UNIVERSIDADE FEDERAL DE SANTA CATARINA
CURSO DE PÓS-GRADUAÇÃO EM CIÊNCIAS DA COMPUTAÇÃO

Aplicação de Desempenho para Validar a Gerência Pró-ativa de Redes

por
Analúcia Schiaffino Morales De Franceschi

Dissertação submetida à Universidade Federal de Santa Catarina
para a obtenção do grau de Mestre em Ciência da Computação

Prof. Dr. Carlos Becker Westphall
orientador

Florianópolis - SC
Fevereiro, 1996.

APLICAÇÃO DE DESEMPENHO PARA VALIDAR A GERÊNCIA PRÓ-ATIVA DE REDES

ANALÚCIA SCHIAFFINO MORALES DE FRANCESCHI

Esta dissertação foi julgada adequada para a obtenção do título de

MESTRE EM CIÊNCIA DA COMPUTAÇÃO

na área de concentração de Sistemas de Computação, sub-área Gerência de Redes (Redes de Computadores) e aprovada em sua forma final pelo Programa de Pós-Graduação em Ciência da Computação.

Orientador

CARLOS BECKER WESTPHALL

Coordenador do Curso

ROGÉRIO CID BASTOS

Banca Examinadora

CARLOS BECKER WESTPHALL (presidente)

OTTO MUNIZ BANDEIRA DUARTE

PAULO JOSÉ DE FREITAS FILHO

BERNARDO GONÇALVES RISO

CATALOGAÇÃO NA FONTE

De Franceschi, Analúcia Schiaffino Morales. Aplicação de Desempenho para Validar a Gerência Pró-ativa de Redes. Florianópolis, CPGCC da UFSC, 1996.
1 V.
Dissertação (mestrado ci. comp.) UFSC - CPGCC, Florianópolis, BR - SC, 1996.
Dissertação: Gerência Pró-ativa de Redes : Monitoração Remota : Gerência de Desempenho : Modelo : Aplicação.

Ao meu marido,

Djalmo Antonio De Franceschi Filho

Agradecimentos

Gostaria de agradecer aos professores e aos colegas do Laboratório de Redes e Gerência (LRG), principalmente às amigas Selena Schönberger e Mirela Sechi Moretti Annoni Notare, que ao longo do curso contribuíram para a realização deste trabalho.

Aos professores integrantes da banca examinadora pela apreciação do trabalho.

Aos pesquisadores da *Delft University of Technology*, na Holanda, principalmente a *Richard Kooijman*, pela contribuição científica e fornecimento do monitor de redes.

Em especial, agradeço ao meu orientador, *Carlos Becker Westphall*, pelo empenho e dedicação a este trabalho e às suas atividades acadêmicas.

Sumário

Lista de Tabelas

Lista de Figuras

Lista de Siglas

CCITT	*Consultative Commitee on International Telephony and Telegraphy*
CRC	*Cyclic Redundancy Code*
CSMA/CD	*Carrier Sense Multiple Acess with Collision Detection*
CLTS	*Connection-less Transport Service*
CPGCC	*Curso de Pós-Graduação em Ciência da Computação*
FDDI	*Fiber Distribution Data Interface*
FTP	*File Transfer Protocol*
ICMP	*Internet Control Message Protocol*
IEC	*International Electrotechnical Commitee*
IETF	*Internet Engineering Task Force*
IP	*Internet Protocol*
ISO	*International Organization for Standardization*
ITU-TS	*International Telecommunications Union Telecommunication Standardization Sector*
LAN	*Local Area Networks*
MIB	*Management Information Base*
OSI	*Open Systems Interconnection*
PROTEM-CC	*Programa Temático Multi-institucional em Ciência da Computação*
RMON	*Remote Monitoring Network*
SNM	*SunNet Manager*
SNMP	*Simple Network Management Protocol*
TCP	*Transmission Control Protocol*
UDP	*User Datagram Protocol*

Resumo

Com o aumento significativo do uso de redes de computadores e de telecomunicações, torna-se necessário realizar um gerenciamento mais eficiente. Neste contexto, vem sendo estudada a gerência pró-ativa de redes. Esta nova concepção visa melhorar a qualidade do serviço oferecido aos usuários. Na abordagem pró-ativa utiliza-se um perfil do comportamento da rede, que pode ser elaborado através de monitorações de dados (ou simulação), para determinar ações preventivas que auxiliam a reduzir e evitar problemas em uma rede. Neste trabalho é apresentado um modelo para o desenvolvimento da gerência pró-ativa. Este modelo é composto por duas entidades principais, uma responsável pela monitoração remota (composto por uma *baseline,* um monitor de rede e um serviço de verificação) e outra pelas ações de gerência (a plataforma de gerência, neste trabalho foi utilizado o *SunNet Manager* da *SunConnect*). Estas duas entidades utilizam um serviço de comunicação para trocar informações de controle.

Através de uma aplicação de desempenho foi possível desenvolver um protótipo com caráter experimental baseado no modelo proposto. Esta aplicação envolve tarefas de monitoração de parâmetros que afetam a performance da rede, entre eles, a distribuição do tamanho dos pacotes, a vazão e taxa de utilização da rede. Finalmente, são apresentados os aspectos mais importantes da implementação, em que ressaltam-se o desenvolvimento da *baseline* e dos serviços de verificação e de

comunicação. São fornecidos ainda, resultados das monitorações realizadas no ambiente de testes e a execução do protótipo.

Abstract

The meaninful growth about computer networks and telecommunications requires a management more efficient to administer such networks. With this context the management was becoming proactive. The proactive new conception is able to improve the quality of services offered to the network users. Based on a network behaviour profile (elaborated by data monitoring or by simulation) makes possible to determine preventive actions, wich will reduce and stop some network troubles before them happen. This work presents a model for developing proactive network management applications. This model has two main entities, one is the remote monitoring entity (it has the baseline, the remote agent and the verification service), the other one makes the management actions (the network management platform, in this work was used the SunNet Management platform). The exchange of control information between the two entities was made by a communication service.

Also, a experimental prototype was developed with a performance management application. This application uses the monitoring of a variety of parameters affecting performance, such as, the packet length distribution, the throughput and the utillization rate. The results of monitorings in the test environment also was provided. Finally, the main aspects of prototype implementation was presented, such as the baseline, the verification service and the communication service development.

Palavras-chave

Gerência pró-ativa,
gerência de redes de computadores,
sistemas distribuídos,
monitoração de sistemas,
agente remoto,
avaliação de desempenho,
degradação da qualidade de serviço.

Capítulo 1

Introdução

1. Introdução

As comunicações têm exercido importante papel na história universal. Muitas atividades nem mesmo funcionariam sem comunicação. A Bíblia narra a história da construção da Torre de Babel. Esta torre deveria ser erguida até o céu. O trabalho corria bem, e nunca tinha sido visto uma tarefa progredir de forma tão rápida. Segundo a Bíblia, no entanto, como uma forma de castigo divino por se tentar alcançar o céu, as pessoas engajadas na atividade passaram a falar línguas diferentes. Sem comunicação, tornou-se impossível continuar o trabalho em grupo e as atividades foram suspensas.

Nos últimos anos, as redes de computadores e telecomunicações têm sido indispensáveis nas atividades desempenhadas em organizações nacionais e internacionais. Muitas tecnologias foram projetadas tornando necessária uma gerência que forneça mecanismos de controle e administração eficientes.

Os sistemas de gerência foram projetados para monitorar as informações e auxiliar na solução dos problemas manifestados nas redes. Tais sistemas são baseados na abordagem reativa, na qual o estado ativo é retornado depois de solucionar o problema na rede (ocorre um problema, o sistema de gerência identifica e alerta o gerente que fica encarregado de solucioná-lo). Em contrapartida, vem sendo discutida uma nova abordagem, a pró-ativa.

No mercado internacional, já existem esforços conjuntos para a realização de atividades de gerência pró-ativa. Alguns fornecedores de equipamentos de redes, tais como, analisadores de protocolos e monitores de rede, começaram a embutir em seus produtos mecanismos para permitir tal gerenciamento. A maioria dos monitores de redes utilizam a RMON MIB para a obtenção de informação (são exemplos produtos dos seguintes fabricantes:"*...Axon Networks Inc (Watertown, Mass.), Frontier, Metrix Network Systems Inc. (Nashua, N.H.), Protools and Telecommunications Techniques Corp. (Germantown, Md.), todos estes já estenderam o uso para redes Token Ring, e a Protools tem adaptado a RMON para redes FDDI para transmissões de banda larga...*" [JAN93]). Até mesmo, empresas de desenvolvimento de produtos para redes locais

estão despertando para este mercado (*"...A Novell tem desenvolvido uma série de módulos chamados NLMS (Netware Loadable Modules) que suportarão a RMON e substituirão o hardware da LANtern por software..."* [JAN93]). Em [JAN93], ainda é relacionada uma série de produtos que fornecem facilidades (de alguma forma) para o desenvolvimento da gerência pró-ativa.

A importância de adotar a abordagem pró-ativa nas atuais plataformas de gerência pode ser explicada pela necessidade de ferramentas mais eficientes na administração das redes de computadores e telecomunicações. Os sistemas de gerência disponíveis no mercado atualmente, entre eles o *SunNet Manager (da SunConnect)*, o *NetView 6000 (da IBM)* e o *OpenView (da HP Hewllet-Packard)*, foram projetados para monitorar os recursos e informar situações de erros que já tenham afetado o funcionamento da rede.

A abordagem pró-ativa mantém o funcionamento normal da rede baseando-se em um perfil do comportamento da mesma. Isto é, preserva o estado ativo da rede, evitando ou solucionando os problemas antes que eles aconteçam. Neste sentido, estão sendo desenvolvidos trabalhos de pesquisa associados a Inteligência Artificial na Universidade Federal do Rio Grande do Sul, os quais enfatizam as correções automáticas com o auxílio de sistemas especialistas [ART94][ROC94].

Este trabalho apresenta um modelo para o desenvolvimento de aplicações para a gerência pró-ativa de redes. Este modelo é composto por duas entidades principais, situadas em segmentos distintos de uma rede. Uma entidade é o **módulo de gerência pró-ativa** e a outra é a **plataforma de gerência** de redes.

O **módulo de gerência pró-ativa** é responsável por realizar a monitoração remota de um segmento da rede, identificando através de consultas à *baseline* (perfil do comportamento da rede) as tendências de aparecimento de problemas na rede. Ao detectar uma tendência, este módulo envia uma notificação para a estação de gerência. Esta notificação é enviada através de um serviço de comunicação entre as duas entidades.

A **plataforma de gerência** recebe a notificação com informações que indicam a possibilidade de degradação da rede. A partir daí, realiza as ações de controle necessárias para evitar que o problema aconteça e impedir a desativação do sistema.

Com o objetivo de testar o modelo proposto foi especificada uma aplicação na área de gerência de desempenho. Entre as funções desta aplicação destacam-se a monitoração de alguns parâmetros que afetam a performance, e a manutenção da qualidade do serviço oferecido pela rede. Além disso são apresentadas também, sugestões para o desenvolvimento de novas aplicações, [DEF95c] apresenta a especificação formal de um agente para a gerência pró-ativa utilizando a TDF LOTOS (Técnica de Descrição Formal LOTOS - *Language of Temporal Ordering Specification).*

A aplicação de desempenho foi desenvolvida com o uso de um monitor de redes e variáveis da RMON MIB *(Remote Monitoring Network Management Information Base).* Estas variáveis fornecem meios para monitorar a performance da redé. Dentre os grupos de objetos da RMON MIB que foram utilizadas pela aplicação destacam-se o Grupo *Statistics,* o Grupo *History,* o Grupo *Hosts* e o Grupo *Traffic Matrix.*

Finalmente, são apresentados os aspectos da implementação do protótipo. Em especial, o desenvolvimento da *baseline,* a implementação do serviço de verificação e a implementação do serviço de comunicação entre as entidades do sistema através de *sockets* (que são recursos para a intercomunicação de processos em sistemas distribuídos).

1.1 Objetivos

A meta principal do trabalho desenvolvido é viabilizar a gerência pró-ativa de redes com o auxílio de uma aplicação de avaliação de desempenho. Através da utilização de ações corretivas de gerência é possível impedir que equipamentos e serviços da rede funcionem em modo degradado. Além disso, através desse trabalho é possível, também, criar novas aplicações, tal como, tratar o congestionamento do tráfego das redes, através da análise do perfil da rede e reconfiguração de seus recursos.

Dentre os objetivos do presente trabalho, destacam-se:

- analisar, através de um protótipo, a viabiliade da construção de uma avaliação de desempenho para a gerência pró-ativa de redes;
- efetuar monitorações do ambiente de testes, gerando exemplos práticos e teóricos sobre a construção da gerência pró-ativa;
- criar um modelo que permita o desenvolvimento da gerência pró-ativa;
- verificar aspectos de implementação, tais como: a importância do uso da simulação para a implantação de uma gerência pró-ativa, ou então; o uso de recursos oferecidos por uma plataforma de gerência (neste caso, *o SunNet Manager*) para efetuar a comunicação entre as entidades do sistema proposto.

1.2 Organização do trabalho

O presente trabalho está organizado em sete capítulos. O Capítulo 2 apresenta aspectos da gerência de redes, abordando os modelos de gerência da OSI/ISO (*Open Systems Interconnection/International Organization for Standardization)* e da *Internet.* Também são apresentados aspectos de gerência de desempenho, gerência pró-ativa e monitoração de sistemas distribuídos. O Capítulo 3 aborda a avaliação de desempenho em sistemas de computação e redes de computadores. O Capítulo 4 descreve a aplicação de desempenho para a gerência pró-ativa e o ambiente de testes. No Capítulo 5 apresenta-se o modelo e a elaboração do protótipo para o desenvolvimento de aplicações para a gerência pró-ativa de redes. Os aspectos de implementação do protótipo experimental são descritos no Capítulo 6. Em acréscimo, apresentam-se a Conclusão, as Referências Bibliográficas, o Índice Remissivo e o Glossário com termos e abreviações citados no corpo do trabalho.

Capítulo 2

Gerência de Redes

2. Gerência de Redes

A gerência de redes engloba métodos para planejar, configurar, controlar, monitorar, corrigir falhas e administrar as redes de computadores. O propósito da gerência é fornecer suporte aos provedores e usuários de forma que a rede e seus componentes possam ser utilizados com eficiência, respeitando assim, a qualidade de serviço oferecida pela rede.

2.1 Modelo de gerência OSI (*Open Systems Interconnection*)

A necessidade de uma tecnologia de rede não proprietária tem sido amplamente reconhecida por várias organizações, incluindo a ISO/IEC (*International Organization for Standardization/ International Electrotechnical Commitee*). Em 1979, a ISO em atividade conjunta com o CCITT (*Consultative Commitee on International Telephony and Telegraph,* atualmente ITU - *International Telecommunications Union*), desenvolveu um modelo de referência para Interconexão de Sistemas Abertos (OSI - *Open Systems Interconnection*). Desconsiderando questões políticas sobre o desenvolvimento do modelo OSI, ele possui um profundo impacto na terminologia de redes [ROS91].

A gerência de redes no modelo de referência OSI é formada pelo modelo informacional, o modelo estrutural e o modelo funcional. O **modelo informacional** determina como a informação deverá ser armazenada, descrevendo as características da MIB (*Management Information Base*) e da estrutura de armazenamento MIT (*Management Information Tree*). O **modelo estrutural** está baseado no paradigma gerente-agente. O gerente solicita operações ao agente, enquanto o agente é um processo que coleta dados dos objetos gerenciados e envia-os ao gerente. O agente pode ainda, enviar notificações ao gerente, caso algum evento anormal ocorra. O **modelo funcional**, por sua vez, organiza a gerência através de grupos de funções. Ele é composto por: *gerência de falhas, configuração, contabilização, desempenho e segurança.*

A Figura 2-1 apresenta a arquitetura do modelo de referência OSI que contém sete níveis de funcionalidades conceituais [COM88].

Figura 2-1 Níveis conceituais do modelo de referência OSI.

2.1.1 Áreas funcionais da gerência de redes

Dentre os objetivos da gerência de redes, ressalta-se o de reduzir o número de problemas em uma rede. Uma vez que os problemas ocorram, a gerência deve minimizar e conter seus danos. O modelo de gerência OSI definiu um modelo funcional classificando a gerência de redes em cinco principais áreas, tal modelo serve para a classificação da gerência de redes em geral [ROS91][ROS94]:

- a **gerência de desempenho** é uma das principais preocupações na administração de uma rede. Inclui o controle e a análise de medidas de performance da rede, tais como, taxa de erros, vazão (*throughput*), tempo de resposta e taxa de utilização da interface. Envolve também, os custos gastos para o funcionamento eficiente da rede;
- a **gerência de configuração** é responsável por detectar e controlar o estado da rede (tanto para configurações físicas quanto lógicas);
- a **gerência de falhas** é responsável por detectar anomalias no comportamento da rede, isolar os problemas e tentar controlá-los. A gerência de falhas normalmente trata as ocorrências de eventos e notificações quando eles surgem na rede;

- a **gerência de contabilização** é responsável por coletar e processar informações relacionadas ao consumo de recursos na rede;
- a **gerência de segurança** é responsável por controlar o acesso de recursos da rede através de técnicas de autenticação e políticas de utilização e autorização (tipos de acesso) em uma rede.

2.2 Modelo de gerência Internet

A arquitetura *Internet* é formada por cinco níveis conceituais como mostra a Figura 2-2 [COM88].

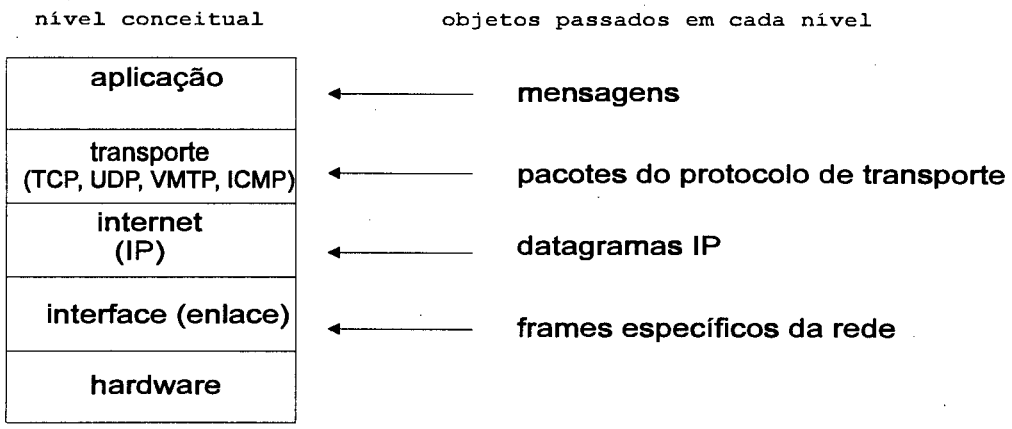


Figura 2-2 Níveis conceituais da arquitetura *Internet*.

- **nível de aplicação**: neste nível os usuários utilizam programas de aplicação fim-a-fim que acessam a *Internet*. Uma aplicação interage com o protocolo do nível de transporte para enviar ou receber dados. Cada programa de aplicação escolhe sua própria forma de transmitir os dados, ou por seqüência de mensagens ou fluxo de *bytes*. Qualquer uma das formas é passada ao nível de transporte.
- **nível de transporte**: o nível de transporte deve fornecer a comunicação entre um programa de aplicação e outro. Este nível pode também fornecer transporte confiável, assegurando que os dados chegarão em seqüência e sem erros. No entanto, é preciso que o lado receptor envie reconhecimentos de volta ao transmissor, e que este retransmita pacotes (terminologia da ISO) perdidos.

- **nível *Internet***: trata a comunicação de máquina a máquina. Aceita uma solicitação para enviar um pacote do nível de transporte com a identificação da máquina onde o pacote deve ser entregue. Este nível encapsula o pacote em um datagrama IP, preenche os dados do cabeçalho, utiliza um algoritmo de roteamento para determinar se a entrega do pacote é direta, ou se deve ser enviada através de um *gateway* (roteador), e passa o datagrama à interface de rede apropriada para transmissão. Este nível trata também dos datagramas recebidos, verificando sua validade, retirando o cabeçalho e utilizando o algoritmo de roteamento para decidir se o datagrama deve ser processado localmente ou ser passado adiante. Para os datagramas endereçados a máquina local o *software* do protocolo *Internet* deve selecionar o protocolo de transporte que irá tratar o pacote. Finalmente, este nível envia mensagens ICMP (*Internet Control Message Protocol*) quando necessário e trata todas as mensagens ICMP que são recebidas pela máquina local.

- **nível de interface de rede**: o protocolo *Internet* de nível mais baixo compreende o nível de interface de rede. É responsável por receber datagramas IP e transmiti-los sobre uma rede específica.

A Figura 2-3 [COM91] apresenta o caminho de uma mensagem que passa do nível de aplicação de um *host* A a um *host* B. Cada nível *n* de B recebe exatamente o mesmo objeto que passou pelo nível *n* de A.

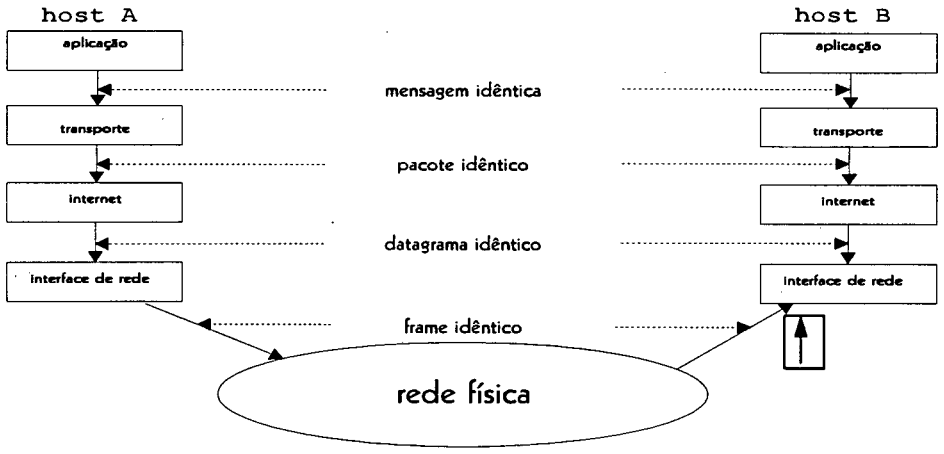


Figura 2-3 Como uma mensagem de um *host* A chega a um *host* B.

2.2.1 Protocolo SNMP (*Simple Network Management Protocol*)

Segundo Comer [COM91], o SNMP auxilia os gerentes de rede a localizar e corrigir problemas em uma rede TCP/IP. É um protocolo assíncrono de requisição e resposta (*request/response*). Isto significa que a entidade não precisa esperar por um resposta depois de enviar a mensagem. A estrutura informacional está baseada em gerentes, agentes, objetos gerenciados e base de informação de gerência (MIB). Os gerentes executam uma chamada de um cliente SNMP na máquina local para contactar um ou mais servidores SNMP que executam em máquinas remotas.

As primitivas (ou operações) de serviço disponíveis no SNMP, são: *get-request; get-next-request; set-request; trap.*

A Figura 2-4 [ROS91] ilustra a operação *get* do protocolo SNMP em dois ângulos, o segundo ressaltando a sua despreocupação com o sincronismo, pois a entidade receptora (o agente) não precisa enviar um reconhecimento (avisando que a solicitação foi recebida) à entidade transmissora (o gerente). Das operações, apenas a operação *trap* é unidirecional, neste caso do agente para o gerente. As demais operações são bidirecionais, o gerente envia uma solicitação para o agente e aguarda a resposta, que parte do agente para o gerente.

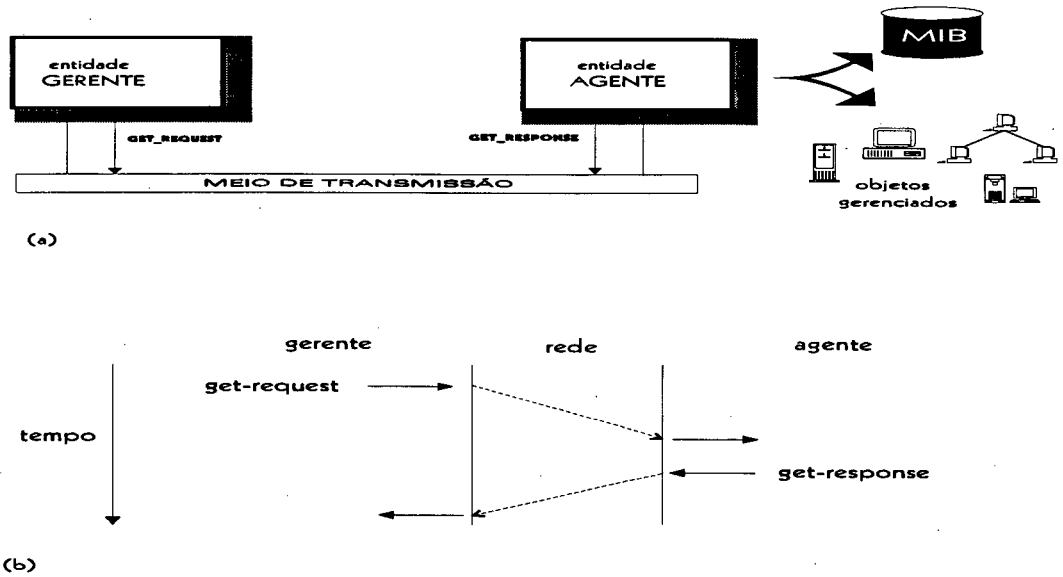


Figura 2-4 (a) a operação get do SNMP, relação entre gerente, agente e objetos gerenciados; (b) o assincronismo da operação get.

O SNMP é conhecido como um protocolo independente de transporte, isto é, ele pode ser definido em vários mapeamentos de transporte, tais como, no barramento *Ethernet*, no protocolo UDP, ou ainda, no OSI CLTS (*Open Systems Interconnection - Connection-less Mode Transport Service*) [ROS91][ROS94].

Todos os tipos de mapeamentos possuem uma coisa em comum, ou seja, as instâncias das mensagens SNMP são transmitidas através da rede por um processo chamado **serialização.** Isto permite que os dados sejam codificados como seqüência de *bytes* para transmissão. Quando tal seqüência de *bytes* é recebida, ela é convertida para a estrutura de dados semanticamente idêntica à que foi emitida [ROS91][ROS94].

O mapeamento mais comum é sobre o UDP (*User Datagram Protocol*). No caso de uma transmissão de uma entidade SNMP, como por exemplo, um gerente. Ele serializa a mensagem SNMP e envia como um único datagrama UDP para o endereço de uma entidade SNMP receptora, um agente, por exemplo. O formato do datagrama é ilustrado pela Figura 2-5 [COM91][ROS91][ROS94].

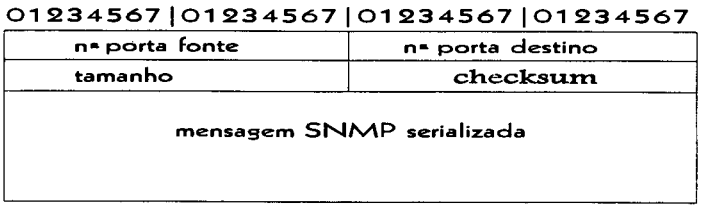


Figura 2-5 Formato do pacote UDP.

O mapeamento do protocolo SNMP sobre o protocolo UDP, no caso de uma rede *Ethernet,* normalmente, através das portas de comunicação 161 e 162 [ROS91][ROS94]. O endereço de transporte consiste de um endereço IP e uma porta UDP. Todos os agentes SNMP escutam a porta 161. Se a mensagem contém uma notificação (uma operação *trap*), o processo que recebe, escuta na porta 162. Por convenção, todas as respostas são enviadas de acordo com o endereço das portas fonte e destino, que são copiadas das requisições correspondentes.

No caso de um *get-request,* por exemplo, enviado de uma porta 1094, terá o *get-response* correspondente enviado para a porta 1094 a partir da porta 161. A Figura 2-6 ilustra a situação de forma genérica, com os agentes escutando a porta 161

e aguardando por alguma mensagem. No momento que chega alguma solicitação direcionada a um determinado agente SNMP (porta-destino 161 e porta-fonte 1094), ele realiza a operação e envia a resposta para a porta que originou a mensagem (neste caso, porta-fonte 161 e porta-destino 1094).

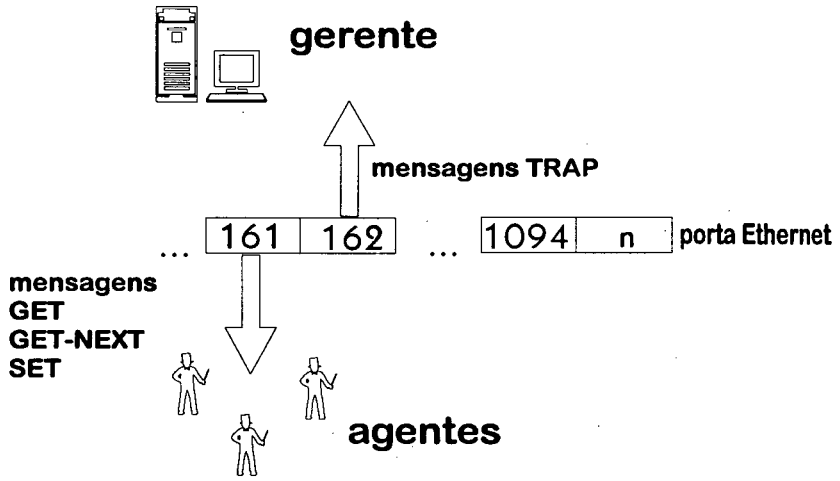


Figura 2-6 Mapeamento do protocolo SNMP através do protocolo UDP.

2.2.1.1 Protocolo SNMP versão 2

Um protocolo de gerência é utilizado para trocar informações de gerência entre agentes e gerentes. As operações do protocolo são transportadas em um quadro administrativo que define tanto políticas de autenticação quanto de autorização. O protocolo de gerenciamento SNMP (*Simple Network Management Protocol*) versão 2, surgiu para suprir algumas necessidades não abordadas pela versão anterior (discutida na Seção 2.2.1).

Uma entidade SNMPv2 pode operar tanto como agente, como gerente. Ela age como agente quando realiza operações de gerência em resposta a mensagens recebidas do protocolo SNMPv2, ou quando envia notificações (*traps*). Uma entidade assume o papel de gerente quando inicializa operações de gerenciamento pela geração de mensagens SNMPv2. Ou quando ele realiza operações em resposta às notificações recebidas. Uma entidade pode ainda, assumir o papel de agente *proxy*, que aparentemente age como um agente comum, porém satisfaz solicitações de gerência agindo como gerente de uma entidade remota [CAS93].

Nessa nova versão do protocolo SNMP foram introduzidas duas novas operações de gerenciamento [CAS93]:

- a operação *GetBulkRequest* é utilizada para solicitar a transferência de um volume de dados maior do que uma operação *GET* permite, ao invés de um objeto de cada vez, pode ser obtido um conjunto de objetos em uma única operação;
- a operação *InformRequest* é gerada e transmitida por um gerente (uma entidade de aplicação SNMPv2) quando precisa enviar uma notificação a uma outra entidade (outro gerente) sobre uma determinada informação da MIB.

2.2.2 O ambiente de gerência do SNM

O *SunNet Manager* (SNM), da *Sun Microsystems*, é um dos sistemas de gerência para ambientes TCP/IP mais utilizados, porque ele agrada tanto a usuários menos exigentes quanto aos mais avançados. Ele permite um certo grau de heterogeneidade porque oferece recursos para gerenciar dispositivos SNMP. Ele é formado por um conjunto de ferramentas que ajudam a gerenciar vários elementos da rede. O ambiente é composto por uma interface gráfica de apresentação da topologia da rede representada por figuras. Tal interface permite a interação dos usuários com os elementos que compõem a rede através da manipulação dos respectivos ícones e uma biblioteca de serviços.

O SNM é baseado no paradigma gerente/agente. O gerente é o processo disparado pelo usuário, que envia as operações e recebe notificações de gerência. O agente é o processo que coleta os dados dos objetos relatando-os ao gerente (respondendo às operações solicitadas pelo gerente e emitindo notificações que refletem o comportamento dos objetos). Este modelo utiliza a técnica de *polling*, na qual o gerente questiona os agentes sobre os resultados das operações solicitadas, além disso, o gerente fica encarregado de selecionar as informações relevantes. Esta técnica permite também que os agentes realizem um TRAP, no qual o agente avisa o gerente que precisa ser submetido ao *poll*.

A biblioteca de serviços é composta por serviços de monitoração e um conjunto de agentes disponíveis aos usuários. O ambiente do SNM é uma plataforma que permite o desenvolvimento de novas aplicações de gerência, através da

construção de novos agentes com o auxílio das API's (*Application Programmer Interfaces*). Após essa construção, tais agentes passam a fazer parte do conjunto original de funções de gerenciamento [SIL93][WES93][ROC94a].

2.2.2.1 Agentes disponíveis

Em primeiro lugar é necessário esclarecer a diferença entre um agente comum e um agente *proxy*. No contexto do SNM, o termo "agente" refere-se a uma entidade que pode agir diretamente a partir do console usando o protocolo proprietário. Novamente no contexto do SNM, um agente *proxy* refere-se a uma entidade que interage diretamente a partir do seu console usando o protocolo do SNM, porém estes agentes fazem isso em favor de uma ou mais entidades que não fazem parte desta arquitetura. Por exemplo, o agente *proxy* SNMP freqüentemente refere-se a um dispositivo que interage com uma estação de gerenciamento baseada no SNMP usando o protocolo SNMP.

A Figura 2-7 ilustra o relacionamento entre gerente e agentes no SNM, onde o processo gerente é disparado apartir da console e pode enviar requisições tanto aos agentes do SNM, quanto aos agentes *proxies* para gerenciar objetos SNMP. Entre os agentes que fazem parte da plataforma SNM estão: *ping, hostif, hostmem2, hostperf, iostat2, ippath, iproutes, layers2, lpstat, rpcnfs, snmp, snmp-mibII, sun-snmp, traffic.*

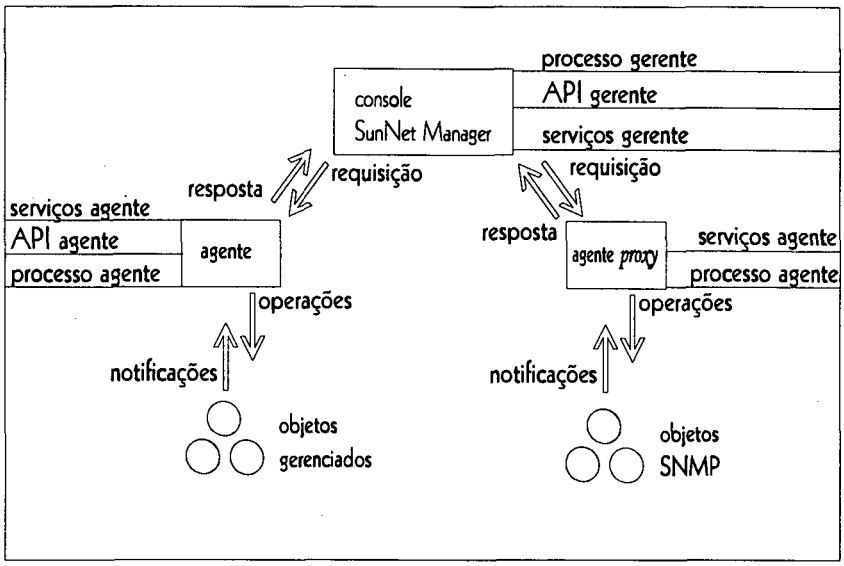


Figura 2-7 Relação gerente, agente e agente *proxy* no SNM.

Para cada um dos agentes são definidos (nos arquivos *.schema*) atributos, tipos de dados, grupos e tabelas que eles podem identificar sobre os objetos gerenciados. Além disso cada um deles pode ser disparado para executar duas operações de monitoração diferentes: monitoração de dados e relatório de alarme de eventos.

Cada agente opera sobre uma coleção de atributos diferentes, tais atributos podem estar organizados em grupos, tabelas, ou apenas atributos simples. Um grupo permite tratar uma coleção de atributos como uma unidade atômica. E uma tabela permite definir uma coleção de atributos de várias instâncias de um componente. Nesse caso, é necessário fornecer uma chave que identifique uma instância da tabela. Como por exemplo, o agente *hostif* possui uma tabela de atributos chamada *if*, onde a chave para selecionar uma instância da tabela é o nome do dispositivo de interface que o usuário deseja gerenciar. Ou então, o agente *ping* possui dois grupos de atributos: *reach* - indica se a máquina de onde o agente foi disparado pode ou não alcançar a máquina fornecida como destino. E *stats* - retorna estatísticas para os pacotes enviados a uma máquina fornecida como destino.

2.2.2.2 *Criação de novos agentes*

O SNM permite a expansão da aplicação de gerenciamento através da criação de novos agentes. Existem alguns trabalhos descritos em [CAR94][SIL93][WES93] e [ROC94a] que apresentam novos agentes SNM desenvolvidos para redes locais. Entre os novos agentes desenvolvidos encontram-se agentes para monitoração de processos, para monitoração de usuários, para monitoração do tráfego entre sub-redes, para monitoração do tráfego de datagramas, para monitoração do uso de microcomputadores interconectados em redes locais, e para monitoração do tráfego dos pacotes TCP/IP, agentes para distribuição de diretórios, para descobrir rotas ativas no domínio *Internet*, para fazer análise qualitativa do tráfego da rede, para fazer análise do congestionamento da rede, entre outros. Estas novas aplicações foram desenvolvidas com o auxílio das API's fornecidas pelo SNM, comandos UNIX, e algumas delas filtram as informações fornecidas por determinados agentes propondo uma nova aplicação.

2.3 Gerência de desempenho

A gerência de desempenho como área funcional da gerência de redes é destinada a monitorar e controlar a performance da rede e seus componentes, obtendo dados estatísticos e agindo sobre os recursos para melhorar a efetividade das atividades de comunicação. Dentro desta função ressaltam-se os seguintes aspectos [NEU93]:

- **Monitoração:** atividades de comunicação com o objetivo de obter dados apropriados para avaliar a performance da rede;
- **Controle:** configurar e modificar parâmetros que controlam a medida dos dados relacionados com a performance em sistemas remotos;
- **Análise:** avaliar os resultados das monitorações com o objetivo de obter novas asserções sobre a performance da rede;
- **Ajuste:** ajustar recursos para melhorar a performance (modificar configurações de recursos ou redistribuir o tráfego da rede, por exemplo).

Entre as métricas de performance que podem ser utilizadas para avaliar o desempenho de uma rede, destacam-se:

- a **carga de trabalho** da interface e dos dispositivos interconectados à rede, isto é, a capacidade dos elementos da rede;
- a **vazão** (ou ***throughput***) da interface, que é a porção da largura de banda que está sendo utilizada pelos dados que circulam na interface;
- e o **atraso de propagação**, que é o tempo que uma informação que está sendo transmitida leva para se propagar no meio de comunicação para evitar colisões.

2.3.1 Principais tarefas da gerência de desempenho

Dentre as principais tarefas da gerência de desempenho, destacam-se [NEU93]:

- **Monitoração da carga de trabalho**: esta atividade engloba dados sobre a utilização de recursos e a taxa de solicitações para a execução do serviço oferecido

pela rede. Além disso deve detectar condições de sobrecargas, informações sobre a capacidade da rede e a taxa máxima tolerada;

- **Monitoração da vazão**: esta tarefa é inerente à monitoração da carga. Como no caso anterior, a informação sobre a vazão máxima é essencial para comparar com a taxa atual e identificar as condições indesejáveis;

- **Monitoração de atrasos de transmissão**: envolve informações sobre tempos de serviço atuais, e máximos aceitáveis, considerando as transmissões de dados ou gerenciamento de conexões (atrasos durante o estabelecimentos de conexões).

As informações utilizadas na gerência de desempenho podem ainda, beneficiar outras áreas funcionais, tais como [PRA95]:

- **Gerência de Falhas**: os dados de performance podem ser utilizados para detectar falhas na rede;

- **Gerência de Configuração**: os dados de performance podem auxiliar na decisão de trocas na configuração da rede;

- **Gerência de Contabilização**: os dados de performance podem ser utilizados para ajustar determinadas taxas de contabilização.

2.4 Gerência pró-ativa

Normalmente, a preocupação com a performance da rede por parte dos administradores inicia somente depois que o serviço já está degradado. Até mesmo os sistemas de gerência convencionais foram projetados para comunicar eventos, alarmes e notificações somente no momento em que os problemas acontecem, ou depois de já terem acontecido. Este tipo de atividade é conhecido como: **gerência de rede reativa**. Portanto, **gerência pró-ativa** é aquela capaz de localizar os problemas antes que eles aconteçam, sejam eles de performance, de falhas, de configuração, de contabilização e de segurança [DAU91].

Até mesmo os fabricantes de alguns equipamentos, tais como, monitores de redes, analisadores de protocolos, *hubs*, entre outros, já se conscientizaram que a gerência pró-ativa é necessária. Assim, começaram a adicionar algum tipo de

inteligência em seus equipamentos para facilitar o desenvolvimento desta nova concepção de gerência. Atualmente, encontram-se disponíveis no mercado *hubs* com a RMON MIB (discutido na Seção 2.4.5) embutida, facilitando assim, a monitoração remota (discutido na Seção 2.4.3) de tais dispositivos.

A gerência pró-ativa pode ser associada a duas outras áreas: inteligência artificial e simulação. Associando-a a inteligência artificial é possível criar sistemas especialistas capazes de diagnosticar os problemas da rede e emitir *troubletickets* automaticamente, adicionando os dados gerados aos arquivos de configuração, por exemplo. Enquanto que, associando a um sistema de simulação, é possível identificar o comportamento do sistema diante de possíveis cenários operacionais e gerenciais da rede. Em ambos os casos é necessário desenvolver uma *baseline* (ou perfil da rede), que apresente os níveis aceitáveis de desempenho.

2.4.1 Objetivos

Dentre os objetivos associados a gerência pró-ativa pode-se destacar alguns relacionados à performance, tais como:

- garantir a qualidade do serviço, mantendo a performance da rede em boas condições;
- ter disponível um perfil da rede, obtendo dados anômalos ou críticos para distinguir o que é normal do que é não adequado;
- compreender padrões de utilização da rede, como por exemplo, usuários mais sobrecarregados, porcentagens de tráfego, etc., utilizando monitores de redes e analisadores de protocolos;
- identificar as ações de controle com o auxílio da simulação, verificando de que maneira a rede poderá suportar mais tráfego, por exemplo.

2.4.2 *Baseline* ou perfil da rede

A função *baseline* ou perfil da rede é uma das partes mais importantes para realizar uma gerência pró-ativa. Para obtê-la é necessário realizar uma atividade de amostragem durante períodos de tempo, identificando a performance normal através de médias e cálculos estatísticos. Desta forma, é possível estabelecer um perfil para

qualquer tipo de rede. No caso da gerência de performance, uma *baseline* pode ser associada a outras funções, tais como, planejamento de capacidade e estimativa de níveis de tráfego com o objetivo de avaliar opções de conectividade [JAN93].

A Figura 2-8 ilustra o desenvolvimento da *baseline* através de técnicas de simulação. Através da modelagem da rede, define-se um conjunto de experimentos sob os diversos cenários operacionais e gerenciais. A partir dos resultados desta modelagem é possível definir o conteúdo da *baseline* e as ações corretivas a serem realizadas durante o gerenciamento.

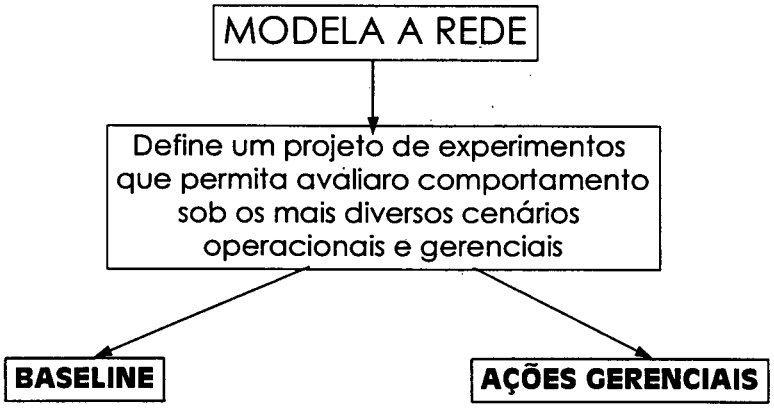


Figura 2-8 Desenvolvimento de *baselines*.

Uma *baseline*, portanto, visa produzir uma caracterização estatisticamente válida do comportamento normal da rede sobre um longo período de tempo; ao contrário de um intervalo específico, como a que atuais sistemas utilizam. Ela deve manter um relatório de níveis de tráfego variado, ou melhor, de diferentes horas do dia ou diferentes dias da semana, ou até mesmo, diferentes dias do mês (no caso de empresas em que os níveis de início e final de mês são normalmente diferentes dos demais).

Além disso, este perfil da rede deve ser verificado e revisado regularmente para manter-se atualizado mediante trocas de padrões de tráfegos que possam acontecer. Por exemplo, uma *baseline* atualizada poderia revelar aumentos estáveis em segmentos de uma rede, indicando a necessidade de uma nova segmentação antes de perder em performance [JAN93]. Algumas das características de uma *baseline* são fornecidas pela RMON MIB (*Remote Monitoring Network Management*

Information Base). A RMON MIB consiste de vários grupos de variáveis, cada uma delas conservando dados de performance sobre os segmentos de uma rede (atualmente está definida para *Ethernet* e *Token Ring*) através dos três primeiros níveis da pilha OSI/ISO. A RMON MIB é discutida na Seção 2.4.5.

2.4.3 Monitoração de sistemas

Uma das principais atividades para a realização da gerência pró-ativa é a monitoração da rede. Na gerência pró-ativa esta monitoração é feita remotamente, isto é, os segmentos de uma rede são monitorados por uma entidade diferente da estação de gerenciamento. Dessa forma é possível diminuir a sobrecarga de tráfego de controle que circula entre um segmento e a estação de gerência. E mesmo que a estação de gerência esteja fora do ar (por exemplo, devido a alguma queda de energia) o segmento permanecerá continuamente gerenciado. A monitoração remota neste trabalho, foi realizada por um agente SNMP que utiliza a RMON MIB (discutido na Seção 2.4.5).

Segundo Mansouri-Samani [MAN93], as informações obtidas durante a monitoração são usadas para tomar decisões de gerência e realizar as ações de controle associadas ao sistema. Portanto, tais informações são utilizadas para avaliar o comportamento e realizar ações corretivas sobre o sistema gerenciado. A monitoração de sistemas neste trabalho, está associada a gerência de redes, que auxilia no desenvolvimento da *baseline* e na aquisição de conhecimento sobre condições anormais dos objetos gerenciados (Figura 2-9).

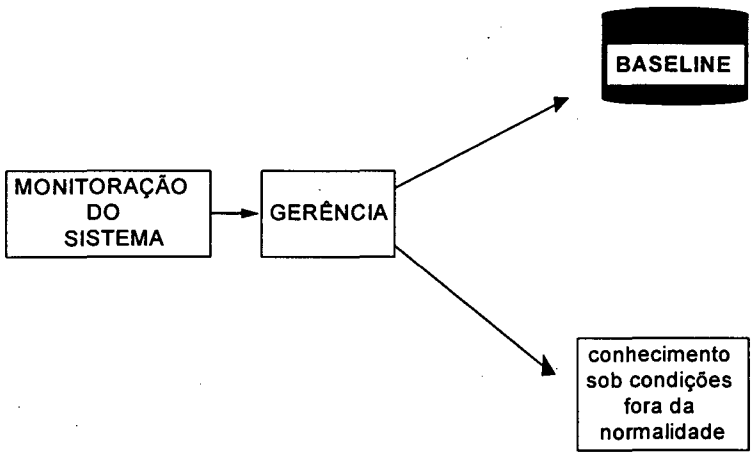


Figura 2-9 Monitoração de um sistema distribuído.

Existem alguns problemas relacionados à atividade de monitoração, dos quais destacam-se:

- Atrasos na transferência de informações, entre o deslocamento de onde as informações foram coletadas até onde serão utilizadas;
- O sistema de monitoração pode competir por recursos com o sistema que está sendo monitorado, modificando assim o seu comportamento.

Em [MAN93] os sistemas de monitoração são classificados em monitores de *software*, monitores de *hardware* e monitores híbridos.

- os **Monitores de *Software*** são programas controlados através de coletores no código fonte com o objetivo de obter as informações de interesse. Normalmente, compartilham recursos necessários com o sistema monitorado. Sua principal vantagem é fornecer dados mais legíveis do que os monitores de *hardware*;
- os **Monitores de *Hardware*** são equipamentos dedicados e são utilizados para detectarem eventos associados a um objeto ou a um grupo de objetos. Sua pincipal vantagem é que não interferem nos resultados do sistema observado e é indicado no uso de sistemas em tempo real;
- os **Monitores Híbridos** são projetados para empregar as vantagens dos monitores de *hardware e software*. Este tipo de monitores têm seus recursos independentes mas também compartilham seus próprios recursos com o sistema monitorado. Sistemas híbridos típicos consistem de dispositivos de *hardware* independentes que recebem a informação de monitoração gerada por coletores de *software* inseridos em objetos dos sistemas monitorados.

2.4.4 Agente ou monitor remoto

O agente ou monitor remoto utilizado é o *Beholder The Next Generation* ou *btng*, o qual foi desenvolvido pelo grupo *DNPAP*, na *Delft University of Technology*, na Holanda. É um *software* de domínio público e está disponível através de *ftp* (*file transfer protocol*) no endereço *dnpap.et.tudelft.nl*.

O *Beholder* ou *btng* (*Beholder The Next Generation*) é um monitor de *software* para tecnologia *Ethernet* capaz de se comunicar através do protocolo SNMP (*Simple*

Network Management Protocol). Este agente é um *software* portável, podendo ser instalado com os sistemas operacionais SunOS, OS/2, Ultrix, Solaris e Linux. Os implementadores deste *software* preocuparam-se principalmente com a portabilidade do sistema, devido aos seus interesses de pesquisa.

Existem, no entanto, algumas diferenças de conformidade entre o RFC1271 [WAL91] e a implementação do *btng*. Por exemplo, o agente não suporta a filtragem dos *status* dos pacotes do Grupo *Filter* da RMON MIB. Apesar desta variável ser muito interessante para se desenvolver uma gerência de falhas, ela restringiria o agente a problemas da interface *Ethernet*.

O agente *btng* suporta os nove grupos de objetos da RMON MIB (discutido na Seção 2.4.5) e será utilizado para coletar tais objetos. Este *software* permite, ainda, a criação de instâncias de tabelas (discutido na Seção 2.4.5.1) da mesma forma como é definido em [WAL91][WAL95]. Na Seção 5.2.2 são exemplificados alguns objetos da RMON MIB e da MIB-II que foram coletados pelo agente BTNG.

2.4.5 RMON MIB

A RMON MIB (*Remote Monitoring Network Management Information Base*) foi desenvolvida para fornecer dados mais detalhados a respeito de performance. Ela é uma extensão do SNMP MIB (*Simple Network Management Protocol Management Information Base)* e surgiu para auxiliar os administradores de redes a detectar, isolar e diagnosticar problemas ou falhas que possam prejudicar a performance e a configuração das redes [DNP94].

A RMON MIB define objetos para uma gerência através de dispositivos de monitoração remota de uma rede. Tais dispositivos são instrumentos que se propõem a gerenciar uma rede, dedicando recursos internos significativos para fazê-lo. Dentro de uma organização podem existir vários dispositivos de monitoração, por exemplo, um para cada segmento da rede, ou então, para cada equipamento ligado à rede, tais como: pontes (*bridges)*, roteadores (*gateways*), hospedeiros (*hosts*) e estações de trabalho (*workstations)* [WAL91][DNP94][WAL95].

A RMON MIB acrescenta novas características aos monitores remotos, tais como:

- contadores adicionais de erros de pacotes;
- histórico e análise estatística;
- matriz de tráfego a nível físico (*Ethernet*);
- poder de filtragem para capturar e analisar pacotes individuais.

A RMON MIB é composta por nove grupos de objetos [WAL91][ERL93][WAL95], são eles:

- o grupo ***Statistics*** fornece estatísticas sobre uma interface *Ethernet*, tais como: pacotes, total de *bytes*, *broadcasts* (endereço de difusão da rede), *multicasts* (endereço de grupos da rede) e colisões em um segmento local, bem como o número de ocorrências de pacotes perdidos pelo agente remoto.
- o grupo ***History*** fornece um histórico das amostras estatísticas coletadas pelo grupo acima mencionado, neste grupo existe um objeto para definir o intervalo para registro das amostras.
- o grupo ***Host*** apresenta uma tabela com todos os hospedeiros (equipamentos com interface *Ethernet*) interconectados na interface. Este grupo armazena todo o tráfego que circula na rede. Nesta tabela, também são armazenadas estatísticas de tráfego tais como: total de pacotes e total de bytes recebidos, *broadcasts* e *multicasts*. Além disso, este grupo fornece uma tabela chamada *Host Time* que fornece a ordem relativa em que cada host foi descoberto pelo agente.
- o grupo ***HostTopN*** é uma extensão ao grupo *host*, ele fornece estatísticas ordenadas dos hospedeiros *(hosts)*, tais como os vinte primeiros equipamentos que enviaram o maior número de pacotes, ou ainda, uma lista ordenada de todas as estações que transmitiram pacotes com erros nas últimas 24 horas.
- o grupo ***Traffic Matrix*** fornece uma matriz de tráfego a nível físico *Ethernet*. Esta matriz mostra a quantidade de pacotes (e *bytes*) enviados e transmitidos entre pares de estações do segmento gerenciado (sendo indexada pelos endereços fonte e destino de cada estação).
- o grupo ***Alarm*** fornece um mecanismo versátil para configuração de limiares (*thresholds*) e intervalos de amostragem para geração de eventos. Isto é

realizado através de um contador mantido pelo agente, tal como no grupo *Statistics*, ou estatísticas definidas na tabela de *hosts*, ou em qualquer combinação de pacotes definida pelo grupo *Filter*. Por exemplo, cruzando um limiar alto pode indicar que existem problemas de performance da rede.

- o grupo ***Filter*** fornece objetos para filtrar informações que podem ser capturadas para o *buffer* do grupo de objetos *packet capture*. O monitor mantém um contador para análises estatísticas que podem ser combinadas para notificar um evento, registrando-o em um *log* ou então, enviando um *trap* SNMP para a estação de gerenciamento.

- o grupo ***Packet Capture*** depende do grupo *Filter*. Os objetos deste grupo permitem que os usuários criem múltiplos *buffers* de captura, permitindo apagar ou parar a coleta quando os *buffers* estiverem cheios. De acordo com a implementação do agente, o tamanho dos *buffers* de captura poderá ser expandido, ou ainda, selecionado o tamanho desejado.

- o grupo ***event*** permite criar entradas no *log* do monitor, registrando todos os eventos ocorridos. Este grupo permite também, que o agente emita *traps* SNMP para a estação de gerenciamento sobre qualquer evento de escolha do usuário. Os eventos podem ser gerados do cruzamento de um limiar ou de uma combinação de um pacote em um filtro (pacote que corresponde as características definidas pelo filtro, por exemplo, um pacote com o endereço destino igual a 80:00:00:20:fd:25). O *log* inclui a hora do dia para cada evento e uma descrição definida pelo fabricante do monitor. A RMON especifica três tipos de *traps* adicionais, além dos cinco definidos pelo SNMP (RFC1157), que são *rising threshold, falling threshold* e *packet match* (removido na nova versão da RMON MIB - RFC1757).

2.4.5.1 Controle de dispositivos de monitoração remota

Devido a complexidade das funções disponíveis nos dispositivos de monitoração remota, freqüentemente são necessárias configurações dos usuários. Como por exemplo, para coletar um determinado tipo de informação é necessário

configurar parâmetros indicando o que deve ser coletado, normalmente a operação só procede depois que todos os parâmetros estiverem configurados [WAL91][WAL95].

Os nove grupos da RMON MIB possuem uma ou mais tabelas com parâmetros de controle, e uma ou mais tabelas de dados, onde são armazenados os resultados das operações. Normalmente, os objetos das tabelas de controle permitem leitura e escrita (operações *get* e *set* do SNMP, discutidas na Seção 2.2.1), enquanto os objetos das tabelas de dados, permitem somente a leitura (operação *get* do SNMP, discutido na Seção 2.2.1).

Para efetuar as operações sobre a RMON MIB e evitar que mais de um usuário (neste caso, um gerente) acesse a mesma informação, é utilizado um mecanismo para adicionar colunas às tabelas de controle (conhecido como *Row-Set*). Este mecanismo permite criar colunas das tabelas de dados através de uma variável de estado encontrada nas tabelas de controle. Existem quatro tipos de estados: *valid(1), create(2), under creation(3)* e *invalid(4)*.

Para criar uma coluna de uma tabela de objetos, os quais serão configurados com valores que o usuário deseja, é necessário atribuir o valor *create(2)* a variável de *status* de uma instância da tabela desejada (uma instância significa uma tabela com um índice definido pelo usuário). Para criar uma instância da tabela *filterTable* da RMON MIB, faz-se:

1. Escolher um índice: neste exemplo, o índice é 10;

2. Linha de comando (em destaque) para realizar a operação SET no monitor utilizado (agente BTNG):

```
%venus echo "filterStatus[10]=2"|snmp-set venus.inf.ufsc.br
secret
```

3. Valor retornado pelo monitor comprovando a criação da coluna 10 da tabela *filterTable*:

```
filterStatus[10]=2
```

A partir daí, é possível configurar os demais objetos da tabela utilizando a primitiva *snmp-set* disponível no monitor remoto. Enquanto isso, o agente atribui à variável de estado o valor 3 (de *under creation*). Após o término da configuração dos

objetos, a variável de estado deverá ser configurada para 1 (de *valid),* para que os objetos configurados possam ser considerados pelo monitor, caso contrário, depois de um certo tempo, o agente considera os objetos inválidos e ignora-os. Da mesma forma, para excluir uma tabela de objetos, a variável de estado deve ser configurada para o valor 4 (de *invalid),* assim o agente desconsidera o conteúdo da tabela.

2.4.5.2 RFC 1757

A RMON MIB tem sido substancialmente alterada deste a sua primeira edição (RFC1271[WAL91] descrito na Seção 2.3.5). Dentre os principais aperfeiçoamentos destacam-se as trocas na definição de objetos do grupo *Packet Capture* (com o acréscimo de um novo *bit* de *status,* e a remoção da notificação *packet match* - combinação de pacotes pelo grupo *Filter*) [WAL95].

Além disso, foram adotadas convenções que auxiliam a compreender os dados armazenados na RMON MIB, as quais foram definidas em conformidade com o padrão IEEE 802.3 (*Institute of Electrical and Electronics Engineers* norma 802.3 para redes locais). Uma das novas convenções adotadas é a respeito dos pacotes considerados bons e dos pacotes descartáveis. De a cordo com o padrão IEEE 802.3, um "pacote bom" é definido como um pacote sem erro com o tamanho do quadro na faixa de 64 a 1518 octetos. E um "pacote descartável" é definido como um pacote com erros (erro de alinhamento CRC) mesmo possuindo um formato adequado (preâmbulo válido), ou tamanho inválido (menor do que 64 octetos ou maior do que 1518 octetos) [WAL95].

Capítulo 3

Avaliação de Desempenho

3. Avaliação de Desempenho

O principal objetivo para realizar uma aplicação de avaliação de desempenho é obter a melhor performance pelo menor custo. Uma avaliação de desempenho é necessária em todo o ciclo de vida de um sistema. No caso de uma rede, inclui seu projeto, compra e venda de equipamentos, novas segmentações, uso, etc. Ajuda, principalmente, a determinar de que maneira a rede está funcionando e se existe algum aperfeiçoamento a ser realizado.

A avaliação de desempenho [JAI91], normalmente, necessita um conhecimento apurado do sistema que está sendo modelado e uma seleção cuidadosa da metodologia, carga de trabalho e ferramentas a serem utilizadas. Existem três técnicas que podem ser utilizadas para a avaliar o desempenho de sistemas: **modelagem analítica, simulação** e **extração**. Algumas das características destas técnicas podem ser visualizadas pela Tabela I.

Tabela I - Critério para a seleção da avaliação de desempenho.

Critério	Modelagem Analítica	Simulação	Extração
1. Estágio	qualquer	qualquer	depois do protótipo
2. Tempo	pequeno	médio	variado
3. Ferramentas	analistas	ling. de computação	instrumentação
4. Confiabilidade	baixa	moderada	variada
5. Custo	pequeno	médio	alto

A modelagem analítica é uma técnica que exige muito do analista, ela pode ser realizada através da teoria das filas, que é utilizada para o desenvolvimento de uma análise de desempenho. Por exemplo, em um sistema de computação vários processos compartilham os recursos disponíveis, tais como: CPU, disco, memória entre outros dispositivos. Normalmente, somente um processo pode utilizar o recurso por vez, os demais que desejam utilizá-lo deverão esperar em filas. A teoria das filas ajuda a determinar o tempo que os processos gastam em várias filas do sistema [FER88][JAI91].

Ao contrário da modelagem analítica, a simulação freqüentemente está mais próxima à realidade. Através da simulação é possível procurar a combinação mais

adequada para os parâmetros de um determinado sistema. Por exemplo, através da simulação poderíamos determinar a vazão (*throughput*) da rede mais adequada e, apartir daí fazer as alterações necessárias para alcançá-la [JAI91].

A extração é um tipo de instrumentação e requer que o protótipo do projeto já esteja desenvolvido para avaliar suas condições de desempenho.

3.1 Simulação

A simulação é uma das ferramentas de análise de sistemas mais poderosas que se tem disponíveis [PED90]. Esta ferramenta pode ser utilizada caso o sistema ainda não esteja concluído (como acontece freqüentemente na fase de projeto de sistemas), e também pode ser utilizada nas fases de planejamento e controle de sistemas. A simulação fornece uma maneira simples para prognosticar a performance ou comparar várias alternativas entre dois ou mais sistemas. Desta forma, mesmo que o sistema esteja disponível para extração de resultados, é preferível utilizar um modelo de simulação, porque ele permite que as alternativas sejam comparadas e testadas sobre uma grande variedade de cargas e ambientes sem interferir no sistema real.

Entretanto, segundo Jain [JAI91], os modelos de simulação podem falhar freqüentemente, produzindo resultados inúteis ou sem significado. Isto poderá ocorrer caso os modelos de simulação não estejam bem elaborados, ou ainda, por falta de conhecimento de técnicas estatísticas e desenvolvimento de *softwares*. Algumas vezes, modelos de simulação podem levar muito tempo para serem desenvolvidos, ocasionando problemas para completá-los. Abaixo são apresentados alguns problemas que devem ser evitados no desenvolvimento de um modelo de simulação:

- Nível de detalhamento não apropriado;
- linguagem de simulação não apropriada;
- modelos sem verificação;
- modelos inválidos;
- condições iniciais tratadas de forma inapropriada;
- simulações muito curtas;

- geradores de números randômicos não adequados; e,
- seleção de amostras iniciais (*seeds*) inapropriadas.

Dentre todos estes problemas destaca-se principalmente a questão da complexidade de modelos. Modeladores iniciantes tendem a carregar demais os seus modelos tentando imitar a realidade. O fato é que na verdade, o principal objetivo de um modelo de simulação não é imitar a realidade, mas **capturar a essência do sistema real sem incluir detalhes desnecessários** (sua capacidade de abstração) [JAI91].

3.2 Métricas de performance

Quando pretende-se desenvolver uma avaliação de desempenho é necessário selecionar os parâmetros relacionados ao tipo de avaliação. Na literatura, são sugeridos diversos tipos de métricas para avaliar a performance de sistemas de computação, apresentados a seguir.

Segundo Jain [JAI91], um sistema pode realizar o serviço corretamente, incorretamente ou recusar a sua realização. Um roteador em uma rede de computadores, por exemplo, pode oferecer o serviço para transmissão de pacotes para destinos específicos em redes heterogêneas. É preciso, neste caso, determinar os parâmetros de performance e o escopo a ser avaliado. Quando chegar um pacote para transmissão, o roteador poderá transmiti-lo corretamente, ou transmiti-lo para um destino errado, ou ainda, o sistema poderá falhar, neste último caso não transmitirá nada. Se o sistema realiza o serviço incorretamente é dito que ocorreu um erro. Por exemplo, se um pacote for entregue parcialmente (fragmentado). Se o sistema não realiza o serviço, é dito que falhou, caiu, ou não está disponível. É muito útil para a gerência de falhas classificar as probabilidades de cada classe de erros. Como exemplo, o roteador pode não estar disponível em 0,01% das chamadas de serviço devido a falhas no processador e em 0,03% devido a falhas de *software*. Se o serviço for realizado corretamente, a performance será medida pelo tempo de resposta, *vazão* (discutida na Seção 3.2.1.1) e utilização dos recursos. O autor classifica as métricas de performance em métricas individuais e globais. Para cada uma dessas

métricas é importante calcular a média e a variabilidade (discutida nas Seções 3.2.1 e 3.2.2).

Segundo Husselbaugh [HUS94], existem três métricas importantes a serem consideradas, além das citadas anteriormente:

- **Tamanho do pacote**: diferentes topologias suportam diferentes tamanhos máximos de pacotes. Em um barramento *Ethernet* por exemplo, o tamanho máximo de um quadro é de 1518 *bytes*, sendo 1024 *bytes* de dados. Já os quadros FDDI (*Fiber Distributed Data Interface*) podem suportar a taxa máxima de 4096 *bytes* de dados. Enquanto em uma *Token Ring* a quantidade máxima de dados por quadro é de 16.384 *bytes* (modo 16Mbps) e 4096 *bytes* (modo 4Mbps).

- **Tempo de acesso ao meio**: este parâmetro é o atraso inerente ao acesso ao meio. Por exemplo, *Ethernet* utiliza CSMA/CD (*Carrier Sense Multiple Access with Collision Detection*). O tempo de acesso do CSMA/CD normalmente é de 5 a 10 vezes mais rápido do que o acesso da passagem de *token* de uma rede *Token Ring*.

- **Largura de banda**: é a faixa de freqüência que um circuito é capaz de transportar. No dia-a-dia refere-se ao limite superior da velocidade em que as informações podem ser transportadas em uma rede.

Para compreender a performance da rede é necessário um modelo apropriado. Husselbaugh [HUS94] propõe um modelo matemático, expresso pela equação:

$$\frac{T'_{rd}}{T_{rd}} = \frac{F.T_{trq} + 2A.T_m + (1 + x[y + P.(1-y)].T_c + P.F.T_{trs})}{(1+x).T_c + 2.T_m + T_{trq} + T_{trs}}$$

Onde:

A = fator de tempo de acesso ao meio;

F = fator de sinalização (freqüência);

P = fator tamanho do pacote;

T_c = tempo total do atendimento ao cliente;

T_m = tempo utilizado para acesso ao meio;

T_{trq} = tempo utilizado para transmitir o pacote de pedido;

T_{trs} = tempo utilizado para transmitir o pacote de resposta;

T_{rd} = tempo total para *baseline* do ciclo de leitura;

T'_{rd} = tempo total utilizado pelo ciclo de leitura modificado;

x = fator de tempo do serviço do servidor de arquivos para a estação; e,

y = porcentagem do tempo total fixado para o serviço.

Esta equação considera todos os elementos de tempo contidos no ciclo de leitura da rede, o qual é composto por:

1) solicitação de leitura *(request)*;

2) tempo para processar a resposta;

3) resposta *(response)*;

4) tempo para formular a próxima pergunta.

Este modelo matemático irá auxiliar na determinação da performance da rede. A partir dos resultados, será possível definir se a performance atual do segmento de testes é aceitável ou não, e até mesmo fornecer uma sugestão para alteração da configuração de tal segmento.

Finalmente, segundo Ferrari [FER78], as principais classes de índices quantitativos de performance para sistemas de computação podem ser analisadas através da Tabela II.

Tabela II - Principais classes de índices quantitativos de performance.

Classe	Exemplos	Definição
produtividade	taxa de *vazão* taxa de produção capacidade taxa de processamento	é o volume de informação útil processada pelo sistema por unidade de tempo
potencial de resposta *responsiveness*	tempo de resposta tempo de *round-trip*	é o tempo decorrido entre a apresentação de uma entrada no sistema e o surgimento de uma resposta correspondente
utilização	utilização a nível de *hardware* (CPU, memória, canal de I/O, etc.) utilização a nível de sistema operacional utilização a nível de *software* (contabilização de FTPs)	é a razão entre o tempo em que uma parte do sistema é utilizada (ou utilizada por uma proposta específica) durante um determinado intervalo de tempo e a duração do intervalo

3.2.1 Métricas individuais

As métricas individuais refletem a utilidade de cada usuário ou de cada equipamento. A vazão e o tempo de resposta são exemplos de métricas individuais.

3.2.1.1 Vazão (ou throughput)

A vazão (ou *throughput*) é a taxa em que os pedidos enviados a um determinado serviço podem ser atendidos. Por exemplo, no caso de uma interface de rede é a quantidade de pacotes transmitidos por unidade de tempo [HAM88][MCK88]. Esta taxa pode ser medida em pacotes transmitidos por segundo (pps) ou *bits* transmitidos por segundo (bps).

Normalmente, a vazão aumenta conforme a carga do sistema. Após uma determinada carga, a vazão estabiliza e em alguns casos pode até decrescer. A vazão máxima sob condições de carga ideal é chamado de **capacidade nominal**, em redes de computadores é conhecido como **largura de banda**. Para calcular a **capacidade útil** do sistema, normalmente, determina-se um tempo de resposta máximo e calcula-se a vazão para este limite.

A razão entre a capacidade útil do sistema e a capacidade nominal (largura de banda) determina a eficiência do sistema. Então, a eficiência de uma rede pode ser calculada através da fórmula abaixo:

$$\textbf{eficiência} = \frac{\textbf{capacidade útil do sistema}}{\textbf{largura de banda}}$$

Por exemplo, se a vazão de uma rede local a 10Mbps é de 0.39Mbps, então a sua eficiência é de 3,9%.

$$\textbf{eficiência} = \frac{\textbf{0.39 Mbps}}{\textbf{10 Mbps}} = \textbf{3,9\%}$$

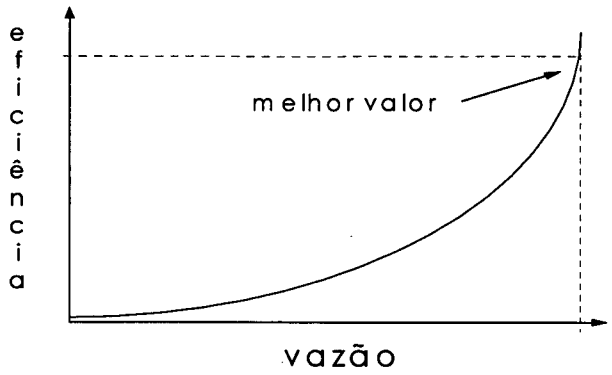


Figura 3-1 Vazão x eficiência.

O gráfico da Figura 3-1 [JAI91] ilustra o relacionamento entre a vazão e sua eficiência. Quanto maior for a vazão maior será a eficiência do sistema, ou seja, a vazão é diretamente proporcional a sua eficiência.

3.2.1.2 Tempo de resposta

O tempo de resposta é o tempo gasto para realizar um determinado serviço. O intervalo de tempo entre a chegada de um pacote ao roteador e sua entrega bem sucedida é um exemplo de tempo de resposta . O tempo de resposta de uma rede, bem como o de qualquer outro sistema de computação, normalmente aumenta quando a carga do sistema também aumenta.

O tempo de resposta da rede pode afetar certos protocolos e interfaces, tais como, NFS (*Network File Server*), *X-Windows* e aplicações cliente/servidor usando mecanismos RPC (*Remote Procedure Call*). Não convém calcular o tempo de resposta utilizando estatísticas do utilitário *ping* ICMP. Os dados obtidos por esse utilitário talvez não sejam os mais exatos, pois dependerão da sobrecarga do dispositivo.

Através do gráfico da Figura 3-2 [JAI91] é possível visualizar o comportamento desta métrica. Quanto menor for o tempo de resposta, mais eficiente será ao sistema, o tempo de resposta é inversamente proporcional a eficiência do sistema.

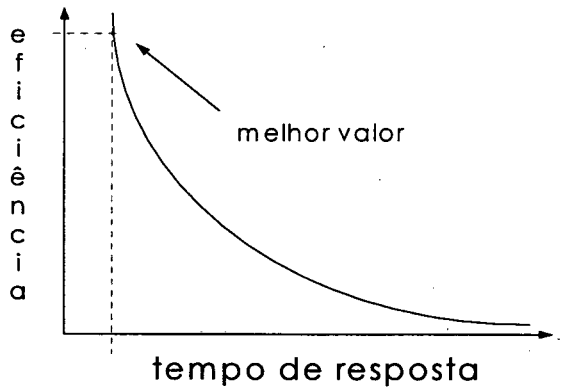


Figura 3-2 Tempo de resposta x eficiência.

3.2.2 Métricas globais

As métricas globais refletem a utilidade geral da rede ou do segmento a ser gerenciado. A taxa de utilização do recurso, a confiabilidade e a disponibilidade são exemplos de métricas globais. As métricas confiabilidade e disponibilidade do sistema, estão diretamente relacionadas a falhas e ao não atendimento do serviço respectivamente, portanto não serão relevantes a este trabalho.

3.2.2.1 Taxa de utilização

A utilização de um recurso é medida através da razão entre o tempo em que o recurso é ocupado, realizando o serviço, e o tempo total decorrido em um determinado período. O período em que o recurso não está sendo utilizado é chamado de período ocioso. Balancear a carga do sistema de forma que nenhum recurso seja super ou sub-utilizado é uma das características mais importantes desta métrica, esta atividade é conhecida como **planejamento de capacidade**.

A taxa de utilização é fornecida em porcentagem. No exemplo do roteador, ela é a porcentagem do tempo em que os recursos do roteador estão ocupados para um determinado nível de carga. O recurso com a maior taxa de utilização é considerado o "gargalo" da rede. Normalmente, é desejável que o valor da taxa de utilização permaneça entre 50 e 75%, um valor baixo desta métrica pode caracterizar uma sub-utilização dos recursos.

O gráfico da Figura 3-3 [JAI91] ilustra o relacionamento entre a taxa de utilização e eficiência. Pela curva do gráfico é possível observar que os valores necessários para obter uma melhor eficiência são no meio do gráfico.

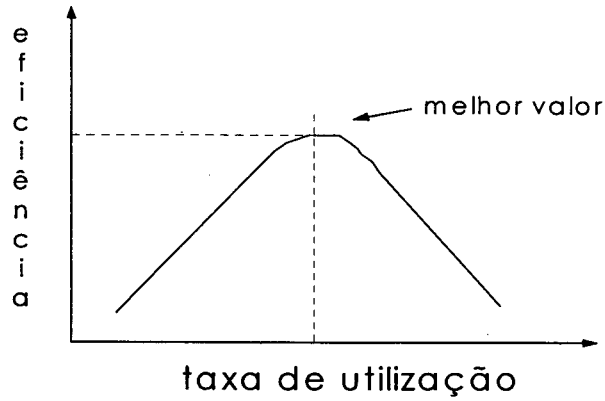


Figura 3-3 Taxa de utilização x eficiência.

Capítulo 4

Aplicação da Avaliação de Desempenho à Gerência Pró-ativa

4. Aplicação da Avaliação de Desempenho à Gerência Pró-ativa

A gerência de performance, ou de desempenho, fornece ferramentas para tornar o funcionamento da rede pró-ativo. Monitores e analisadores de protocolos permitem realizar análises significativas a respeito do tráfego da rede. Através desses dispositivos é possível compreender os padrões de utilização diários da rede, os usuários mais sobrecarregados, as porcentagens variadas dos diferentes protocolos que compõem o tráfego, os gargalos da rede, além de outras informações.

Através da simulação será possível estudar a quantidade de tráfego adicional que a rede pode suportar. Este tipo de exercício pode resultar em benefícios pró-ativos, como por exemplo, determinar a melhor maneira de distribuir os recursos para melhorar a performance. Utilizando os resultados de padrões de tráfego da rede, é possível decidir onde particionar a rede para melhorar a vazão e o tempo de resposta da rede e como os recursos deverão ser alocados.

4.1 Objetivos

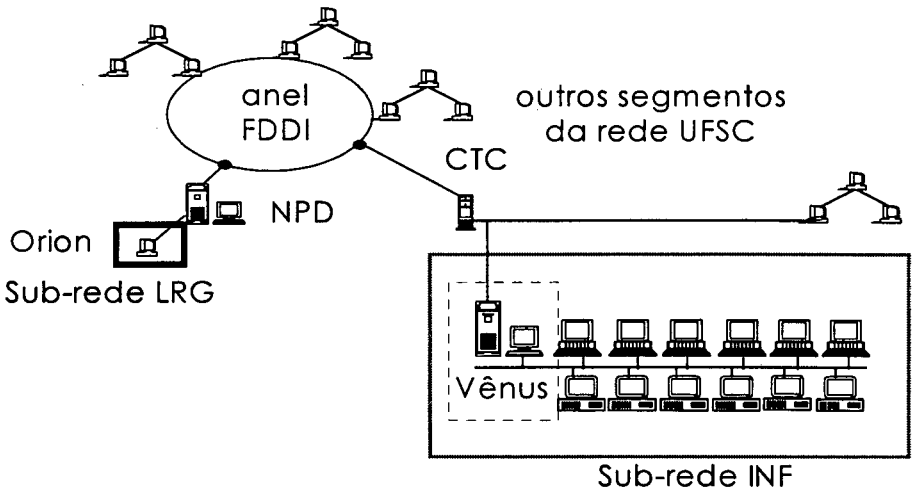
A escolha de desenvolver uma aplicação de avaliação de desempenho para viabilizar a gerência pró-ativa deve-se ao fato da importância que a performance exerce sobre as redes atualmente. A questão da performance é inerente aos investimentos da área de redes e telecomunicações.

Dentre os objetivos desta aplicação destacam-se:

- garantir o melhor desempenho da rede, evitando possíveis problemas que possam degradar a qualidade dos serviços oferecidos;
- desenvolver um perfil do comportamento da rede, incluindo caracterizações do tráfego, taxas de erros, etc;
- manter atualizadas as informações sobre o perfil da rede;
- reduzir os custos mantendo a rede sempre funcionando com a performance o mais próximo possível da ideal.

4.2 Ambinte de testes

O ambiente de testes é composto por duas sub-redes da UFSC. A sub-rede *lrg*, do tipo *Ethernet* (discutido na Seção 4.2.1), faz parte do Laboratório de Redes e Gerência e possui uma estação SUN SPARC 20 (denominada Orion) e está ao Núcleo de Processamento de Dados (NPD da UFSC). E a sub-rede *inf*, também do tipo *Ethernet*, a qual faz parte do Curso de Pós-Graduação em Ciência da Computação, e é composta por uma série de equipamentos, entre eles estações SUN, microcomputadores e impressoras. A estação SUN SPARC 10 (denominada Vênus) é responsável por rotear as mensagens do correio eletrônico para os demais elementos da sub-rede. Atua também como servidor e roteador do segmento. A Figura 4-1 ilustra o ambiente de testes, enfatizando as duas principais estações: a **Orion** e a **Vênus**. O sistema de gerência *SunNet Manager* está localizado na estação Orion, o Agente Remoto e o Módulo de Gerência Pró-ativa (discutido na Seção 5.1) estão instalados na estação Vênus. No Anexo I encontra-se uma ilustração da topologia da rede da UFSC, na qual é possível visualizar o segmento de testes dentro do contexto da rede da universidade.



CTC - Centro Tecnológico
NPD - Núcleo de Processamento de Dados

Figura 4-1 Ambiente de testes.

4.2.1 Ethernet (padrão IEEE 802.3)

Ethernet é o nome dado ao tipo de rede responsável pela transferência de pacotes de redes locais lançada pela XEROX em meados dos anos 70. Um segmento de rede *Ethernet* consiste de um cabo coaxial de aproximadamente ½ polegada e 500 metros de comprimento (para estender o comprimento do cabo é necessário o uso de repetidores). Existe uma resistência que envolve o centro do cabo que previne a reflexão de sinais elétricos. O cabo propriamente dito, é completamente passivo, chamado *ether;* todos os componentes eletrônicos ativos responsáveis pela função da rede estão associados aos computadores ligados à rede. Para ligar um cabo ao outro são utilizados repetidores que transmitem os sinais elétricos de um cabo a outro.

4.2.1.1 Características

É um barramento de difusão (*broadcast)* com velocidade de 10Mbps (existe também o *fast Ethernet* com velocidade de até 100Mbps) com controle de acesso distribuído. É um barramento porque todas as estações compartilham um único canal de comunicação. É de difusão porque todos os *transceivers* (conectores de interface *Ethernet*) recebem a transmissão, isto é, as mensagens de difusão são emitidas a todas as estações interligadas ao barramento.

O controle de acesso é distribuído porque, na maioria das redes, não existe uma autoridade central garantindo o acesso. O esquema de acesso é chamado CSMA/CD (*Carrier Sense Multiple Access with Collision Detection*). No CSMA cada ponto de acesso múltiplo detecta a portadora de sinal que determina quando a rede está ociosa. Cada estação que deseja transmitir uma mensagem, "escuta" o meio para verificar se alguma mensagem está sendo transmitida. Quando nenhuma transmissão é detectada a estação inicia a transmissão. Cada transmissão possui uma duração limitada (tamanho máximo de um pacote) e existe um tempo mínimo necessário em que o meio deve ficar ocioso entre as transmissões. Isto significa que mais de uma estação não pode transmitir a menos que tenha a oportunidade de garantir o acesso ao meio.

O método utilizado para direcionar os pacotes de uma estação a outra ou a um subconjunto de estações é auxiliado por um mecanismo de endereçamento. Cada interface recebe uma cópia dos pacotes, mesmo se forem endereçados a outras máquinas. O *hardware* filtra os pacotes, ignorando aqueles endereçados a outras máquinas, repassando ao hospedeiros (*hosts*) somente os pacotes com o seu endereço de destino. O comportamento do barramento *Ethernet* é ilustrado pela Figura 4-2.

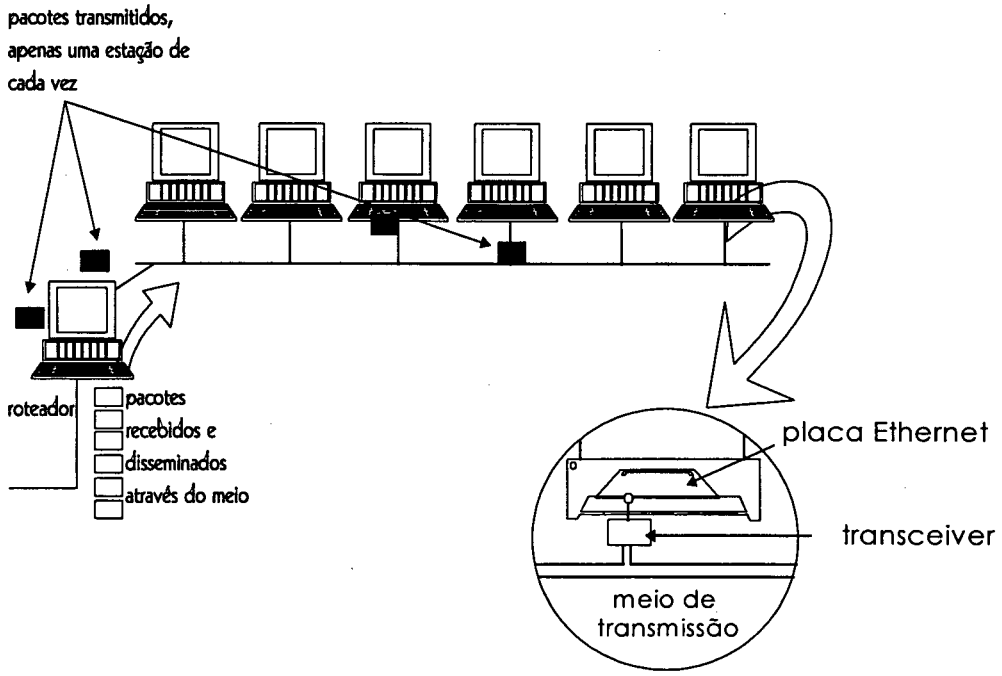


Figura 4-2 Componentes de um segmento de rede Ethernet.

4.2.1.2 Endereçamento

Toda a interface *Ethernet* possui um mecanismo de endereçamento que permite identificar para onde são transmitidos os quadros que atravessam o meio físico. Os quadros são enviados a todas as estações interconectadas ao barramento. Cada estação fica encarregada de filtrar os quadros através da comparação entre o seu endereço físico e o endereço destino de cada quadro [COM88].

O endereçamento *Ethernet* pode ser de três tipos:

- endereço físico de uma interface de rede: é o endereço fornecido pelo fabricante do hardware do equipamento, ele possui 48 *bits* e é chamado de endereço *Ethernet;*
- endereço *broadcast*: por convenção este tipo de endereço (todos os *bits* configurados com o valor 1) é reservado para enviar mensagens a todas estações, simultaneamente;
- endereço *multicast*: fornece uma forma de difusão limitada, onde apenas um subconjunto de computadores na rede possam responder ao endereço *multicast*. Isto é, é um endereço relativo a um grupo de computadores que podem ser alcançados simultaneamente.

Os quadros *Ethernet* possuem tamanhos variados, sendo que nenhum ultrapassa os 1518 *bytes*. O formato dos quadros que atravessam o meio de transmissão a nível físico é ilustrado pela Figura 4-3 [COM88].

64 bits	48 bits	48 bits	16 bits	368-12.000 bits	32 bits
preâmbulo	endereço destino	endereço fonte	tipo de quadro	dados	CRC

Figura 4-3 Formato do quadro *Ethernet.*

O preâmbulo consiste de 64 *bits* com valores alternados entre 0 e 1, que auxilia no recebimento de nodos sincronizados. O CRC (*Cyclic Redundancy Check*) auxilia a interface a detectar erros de transmissão.

4.2.2 Capacidade do *Ethernet*

Os estudos sobre a capacidade do *Ethernet* iniciaram em meados de 1976 (inicialmente, com um protótipo experimental a *3Mbps*) e prolongaram-se até 1988. Desde então, as redes *Ethernets* (com exceção do *fast Ethernet - 100Mbps*) não têm sido modificadas, portanto, tais estudos devem ser considerados. Entre eles, destacam-se:

- **Metcalfe e Boggs, 1976**: No texto original [MET76], é fornecido uma análise de uma rede *Ethernet* experimental a 3Mbps desenvolvida pela XEROX PARC, entre 1973 e 1974. Nesta análise é considerada a vazão (*throughput*) da rede

em função do tamanho dos pacotes e o número das estações. A vazão permanece próximo a 100% para pacotes grandes (de até 512 *bytes*, neste caso), mesmo com o número máximo de estações. Ao contrário, no entanto, para pacotes pequenos, em que a vazão decresce para $1/e$ (cerca de 37%), aproximadamente.

- **Almes** e **Lazowska, 1979**: Realizaram suas análises em uma rede *Ethernet* experimental [ALM79]. O estudo apresenta valores do tempo de resposta em função da carga da rede. Segundo esta análise, para pacotes pequenos (até 576 bytes) o tempo de resposta permance inferior a 1 *milisegundo* para uma carga da rede menor que 75%. Se a carga cresce além deste ponto, o tempo de resposta tende ao infinito. Para pacotes grandes (de 256 *bytes long* de tamanho) a tendência ao infinito surge a uma carga bem maior, tendo um tempo de resposta pior quando a carga da rede é baixa. De acordo com suas análises a peformance da rede é dependente do *slot time* (fatia de tempo que cada estação tem para transmitir).

- **Gonsalves** e **Tobagi, 1986**: Em sua análise foi investigado o efeito de vários parâmetros de performance usando simulação sobre uma rede *Ethernet*. Em particular, foi analisado como a distribuição de estações ao longo do cabo afeta a performance. Estudos anteriores assumiram uma configuração estrela balanceada (todos as estações separadas pelo tamanho máximo de uma rede). Como isto é uma decisão fora da realidade, foram simuladas várias outras configurações. Se as estações estão distribuídas uniformemente ao longo do cabo, ou se estão distribuídas em mais de dois segmentos do mesmo tamanho, as estações localizadas no meio do cabo obtém uma vazão e um atraso ligeiramente melhor do que as estações dispostas no final do cabo. Se os segmentos são de tamanhos diferentes, as estações no segmento maior obtém um serviço significativamente melhor. Isto é porque as estações dentro do segmento realizam o tratamento de colisões rapidamente, da mesma forma que as colisões no segmento, levam mais tempo para tratar as colisões devido a distância maior entre as estações, um problema mais provável dos segmentos menores. Isto é um importante resultado porque um número de análises teóricas assumem explicitamente que o projeto da

rede não afeta a performance. Usuários com aplicações de tempo real podem encontrar alguma posição desfavorável em uma configuração *Ethernet* não usual que pode ocasionar um efeito importante.

Em Tanenbaum [TAN88][TAN94] é apresentada resumidamente, uma análise do desempenho de um segmento de rede *Ethernet* baseado no estudo de Metcalfe e Boggs (1976):

"Em condições de carga alta e constante, com K estações prontas para transmitir, e uma probabilidade constante de retransmissão em cada abertura do canal. É introduzida a seguinte equação:

$$Eficiencia\_do\_canal = \frac{1}{1 + 2.B.L.e / c.F} \qquad (4\text{-}1)$$

*Onde, **F** é o comprimento do cabo, **B** é a largura de banda passante na rede, **L** da velocidade de propagação do sinal, **c** para o caso ótimo de **e** aberturas de contenção por quadro. Quando o segundo termo do denominador é grande, a eficiência da rede será baixa. Mais especificamente, aumentar a banda passante da rede ou a distância (produto **BL**) reduz a eficiência para um certo tamanho de quadro".*

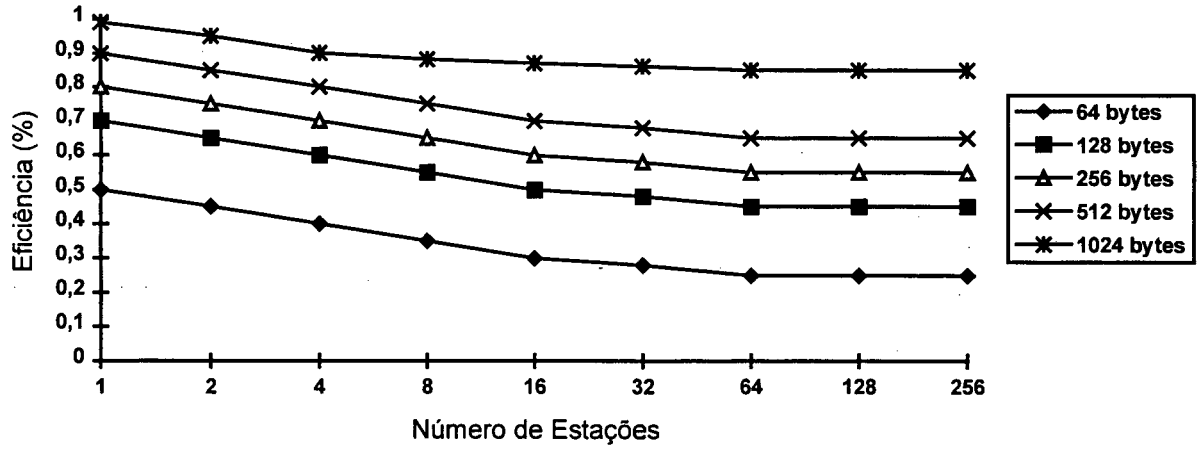


Figura 4-4 - Eficiência do *Ethernet* experimental a 3Mbps considerando o número de estações no barramento e o tamanho fixo dos pacotes.

Na Figura 4-4, a eficiência do canal é traçada versus o número de estações prontas para transmitir, para 2π = 51,2 μs e uma taxa de dados de 10Mbps, usando a

Equação (4-1). Com um tempo de abertura de 64 bytes, não é surpreendente que quadros de 64bytes não sejam eficientes. Por outro lado, com quadros de 1024 bytes e um valor assintótico de e aberturas de 64 bytes por intervalo de contenção, o período de contenção é de 174 bytes e a eficiência de 0,85.

4.3 Problemas que podem degradar a performance

Partes diferentes de uma rede sofrem diferentes tipos de problemas. Por isso, Dauber [DAU91] sugere uma divisão dos componentes de uma rede em quatro categorias:

- ***hardware* da rede:** problemas de conectividades físicas são os mais comuns porque o *hardware* fica exposto ao estresse ambiental e portanto acessível a danos. Dentre eles, destacam-se: quebras de cabo (cabo cortado ou não terminado adequadamente), cabos curtos (podem ser facilmente danificados), quedas em um circuito (um conector vampiro pode ser atingido perdendo contato com o meio de transmissão) e mau funcionamento no circuito de *hardware* da rede (um controlador de interface ruim ou *jabbering tranceiver*). Problemas de cabo podem ser detectados através do uso de um analisador de rede ou TDR (*Time-Domain Reflectometer*). Problemas com circuitos de *hardware* podem ser freqüentemente encontrados examinando o tráfego de erros através de um monitor de rede. Tais problemas também podem ser isolados através de um processo de eliminação.

- **equipamentos de interconexão:** com o aumento do uso das redes, os produtos de interconexão estão se tornando amplas fontes de problemas comuns. Devido a intersecções dentro do padrão de tráfego, problemas significativos poderão ser gerados durante a ocorrência de mau funcionamento na rede. Erros de configuração também são muito comuns com produtos complexos, tais como roteadores, pontes e roteadores. Se os nodos de um lado de um produto de interconexão são afetados, inicia-se uma pesquisa com tal produto para detectar as causas. Deve-se garantir que as filas de processamento não crescerão de forma a escapar do controle do gerenciamento. Neste caso, é importante questionar-se

sobre as últimas alterações que tenham sido efetivadas na rede, e quais seriam os efeitos colaterais que não foram previstos.

- **protocolos de rede:** embora *softwares* de protocolos estejam propensos a erros da mesma forma que quaisquer outros, é somente possível identificá-los e informar aos fabricantes. Incompatibilidades entre *softwares* de protocolos são comuns (interoperabilidade de soluções proprietárias). Analisadores de redes com decodificadores de protocolos embutidos são úteis para detectar uma variedade desses problemas.

- **aplicações de rede:** as aplicações muitas vezes têm erros, porém existem pouquíssimas chances de solucionar esses problemas, a menos que a aplicação tenha sido desenvolvida pela própria organização.

4.4 Aplicação de Desempenho

Com base nos estudos teóricos foi definida uma aplicação de desempenho para viabilizar a gerência pró-ativa de redes. Esta aplicação de desempenho englobou as atividades descritas a seguir (com base no escopo de gerência de desempenho apresentado na Seção 2.3):

- a monitoração da sub-rede *inf* com o objetivo de coletar as informações necessárias para a realização de uma avaliação de desempenho;
- a análise das informações para o desenvolvimento de um perfil do comportamento da sub-rede;
- e o ajuste e controle dos parâmetros analisados.

4.4.1 Parâmetros que afetam a performance

Para o desenvolvimento da aplicação de desempenho proposta foram considerados parâmetros dependentes dos usuários (ver Tabela III). Estes parâmetros

são dependentes dos usuários porque vão variar de acordo com a utilização e a configuração definida pelos usuários da rede gerenciada. Como resultado, obtém-se a vazão, a taxa de utilização da interface, a taxa de colisões, a freqüência de utilização das estações e o fluxo do tráfego da sub-rede.

Em [BOG88] são apresentados alguns parâmetros que afetam a performance de uma rede *Ethernet*. Estes parâmetros estão descritos na Tabela III:

Tabela III - Parâmetros que afetam a performance de uma rede *Ethernet*.

Parâmetro	Descrição	Tipo
Taxa de bit	o *Ethernet* opera a 10Mbits/s , e o atraso máximo de propagação definido é de 464 *bit times* (equivalente a 46,4ms)	operacional
Jam Time	um transmissor quando detecta uma colisão continua a transmitir por mais 32-48 *bit times* para assegurar que os demais equipamentos detectem confiavelmente a colisão	operacional
Slot Time	é a fatia de tempo de aquisição do meio. Deve ser maior que a soma do tempo máximo de propagação mais o tempo máximo de *jam time* (512 *bit times* = 51,2ms)	operacional
Tamanho mínimo dos pacotes	é de 64 octetos (com os 14 octetos do *header* + 4 octetos do *frame check sequence*) Não pode ser menor que o *slot time* para que as colisões possam ser detectadas	operacional
Tamanho máximo dos pacotes	é limitado em 1518 octetos devido ao tamanho dos buffers dos receptores que devem limitar o tempo de retenção Do meio	operacional
Número de estações	a especificação do *Ethernet* limita este valor a 100, e em uma rede multisegmentada o número máximo de estações deve ser de 1024.	operacional
Distribuição do tamanho dos pacotes	segundo a literatura, tem sido observado uma distribuição bimodal, com pacotes nem próximo do tamanho mínimo nem próximo do máximo	dependente do usuário
Taxa de chegada	embora uma rede *Ethernet* permita que uma estação transmita um pacote de tamanho mínimo a cada 67.2ms, a maioria das estações são incapazes de transmitir ou receber mais que poucas centenas de pacotes por segundo. Isto limita a taxa de chegada dos pacotes.	dependente do usuário
Comprimento do cabo	o comprimento do cabo não pode ser maior que o especificado, para que as colisões possam ser detectadas tão logo elas acontecam, o pior caso de propagação do sinal determinado na especificação é de 51.2ms	dependente do usuário

4.4.2 Monitoração da sub-rede

A monitoração da sub-rede *inf* foi dividida em duas etapas, a primeira foi realizada durante o ano de 1995, e a segunda durante a implementação do protótipo. A primeira etapa (realizada antes da implementação do protótipo), está representada pelo diagrama da Figura 4-5, teve como objetivo monitorar a rede e coletar informações (ver utilização do agente remoto, discutido na Seção 5.3) sobre a performance para construir a *baseline* (discutido na Seção 2.4.2).

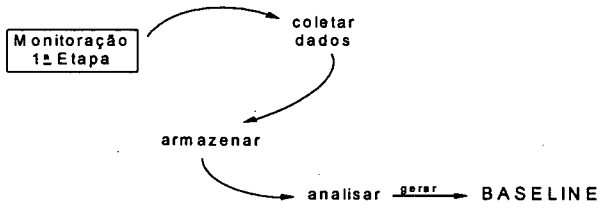


Figura 4-5 Diagrama da primeira etapa da monitoração da sub-rede inf.

A segunda etapa (realizada durante a execução do protótipo), teve como objetivo monitorar a sub-rede para verificar as tendências de degradação de performance, e está representada pelo diagrama da Figura 4-6.

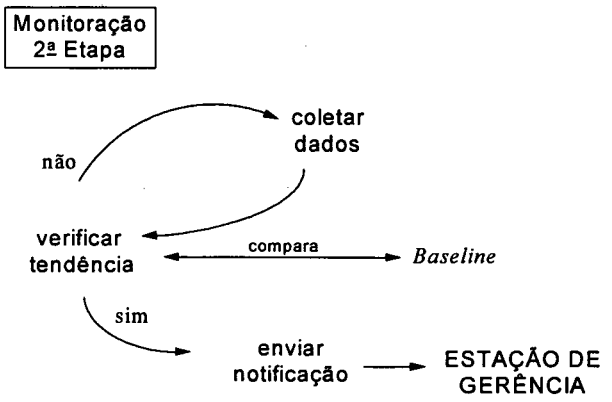


Figura 4-6 Diagrama da segunda etapa de monitoração da sub-rede inf.

As seguintes tarefas compõem a atividade de monitoração da sub-rede *inf*:

- monitoração da distribuição de pacotes;
- monitoração da taxa de transmissão e recebimento de pacotes (e *bytes*);
- monitoração da utilização da interface;
- monitoração da taxa de colisões;
- monitoração do tráfego entre as estações.

4.4.3 Análise das informações

Com o objetivo de avaliar os resultados obtidos durante a monitoração da sub-rede *inf* foi necessário realizar uma atividade de análise. Esta análise teve como produto final o delineamento do perfil da sub-rede sob os aspectos monitorados.

4.4.4 Ajuste e controle

O ajuste e controle dos parâmetros de performance devem ser realizados pelo gerente (ou administrador da rede) quando uma notificação de tendência de degradação for recebida. Futuramente, as ações de ajuste e controle serão realizadas automaticamente com o auxílio de sistemas especialistas (através de bases de conhecimento elaboradas a partir das correções de problemas de performance).

4.4.5 Como a performance é medida

A performance de uma rede, como a maioria dos sistemas de computação, não pode ser avaliada através de uma simples dimensão [BOG88]. Para avaliar a performance da sub-rede *inf* foram consideradas as seguntes métricas:

- a vazão, que é a fração da largura de banda que está atualmente sendo utilizada para transportar os dados;
- a capacidade do canal, que é a vazão máxima suportada pela interface considerando um conjunto de parâmetros (que pode ser o tamanho dos pacotes ou o tamanho da rede);
- a estabilidade da rede - uma rede é dita não estável se a vazão diminui quando a carga aumenta significativamente.

Capítulo 5
Modelo para Gerência Pró-ativa

5. Modelo para Gerência Pró-ativa

Este capítulo descreve o desenvolvimento do modelo e a elaboração do protótipo para viabilizar a gerência pró-ativa de redes. O principal aspecto deste modelo é sua independência de aplicações. Em [NOT96] é apresentado a especificação formal de uma nova aplicação (gerência de tráfego de pacotes) para a gerência pró-ativa de redes com o uso da técnica de descrição formal LOTOS [ISO88]. São exemplos de novas aplicações:

- contabilização de recursos da rede;
- controle de congestionamento da interface;
- análise no roteamento de pacotes;
- controle de acesso à rede.

Em acréscimo, são apresentados os aspectos relacionados a monitoração da sub-rede *inf* da UFSC e a metodologia utilizada para avaliar os resultados obtidos.

5.1 Componentes do Modelo

A partir dos estudos realizados em gerência de redes e sistemas distribuídos foi desenvolvido um modelo para um sistema de gerência de redes pró-ativo, composto por:

1) o **Módulo de Gerência Pró-ativa** é responsável por monitorar e analisar a rede, com o objetivo de verificar as tendências de degradação da rede, conforme a aplicação especificada. Se constatada alguma tendência, deve enviar notificações para a Plataforma de Gerência. Os componentes são a *baseline*, um monitor de rede e um serviço de verificação.

- a ***Baseline*** é um registro do comportamneto normal da rede freqüentemente consultada pelo Serviço de Verificação;

- o **Monitor de Rede** ou **Agente Remoto** (discutido na Seção 2.3.3) coleta as amostras de objetos gerenciados das MIBs em questão (no caso da aplicação de gerência de performance estas MIBs são a RMON MIB e MIB II);

- o **Serviço de Verificação** é responsável por verificar as tendências de degradação da rede, por meio de comparações entre os valores armazenados na *Baseline* e os valores coletados, constantemente, pelo Agente ou Monitor Remoto;

2) a **Plataforma de Gerência** é responsável por receber as notificações e realizar as ações corretivas para impedir a degradação da rede. Atualmente, as ações corretivas estão sendo determinadas pelo gerente (ou administrador da rede). No entanto, pretende-se desenvolver um sistema de correções automático com o uso de sistemas especialistas. Neste caso, quando a Plataforma de Gerência receber alguma notificação (originada da sub-rede monitorada) este sistema será ativado determinando as possíveis soluções para impedir que as falhas aconteçam. Para implementar um sistema deste porte deverá ser desenvolvida uma base de conhecimento com regras baseadas no comportamento da rede.

3) o **Serviço de Comunicação** é responsável por interconectar o Módulo de Gerência Pró-ativa e a Plataforma de Gerência. Este serviço pode utilizar um protocolo de transporte ou o protocolo de gerência SNMP (discutido na Seção 2.2.1).

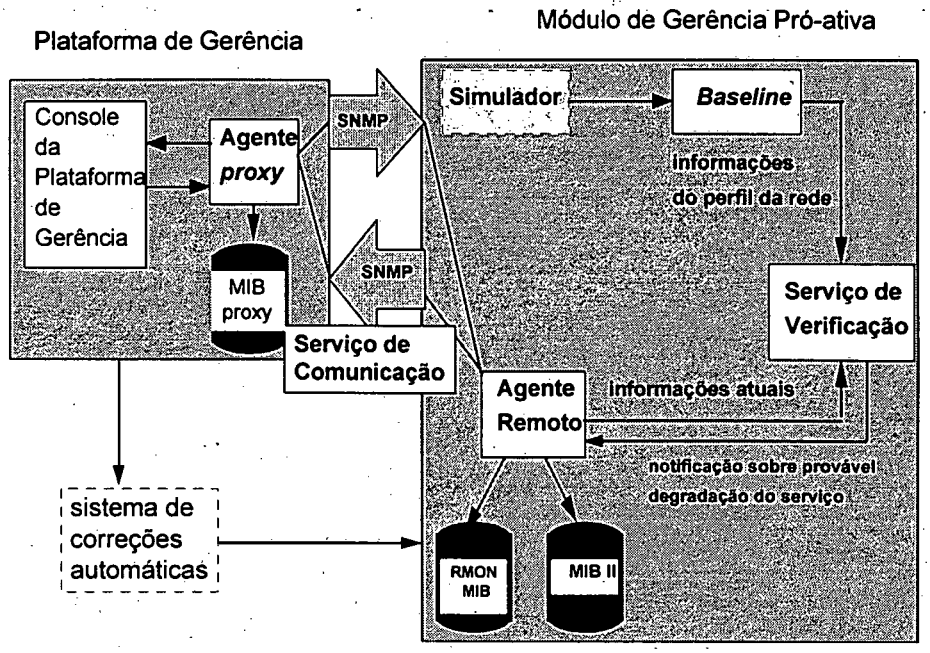


Figura 5-1 Modelo para a gerência pró-ativa de redes.

A Figura 5-1 ilustra os componentes que compõem o modelo para implementar um sistema de gerência pró-ativa (citados anteriormente). A função do

simulador neste modelo é gerar informações relevantes aos aspectos operacionais e gerenciais da rede. Através do uso da simulação é possível determinar as ações de controle a serem realizadas e delinear a *baseline* estatisticamente correta.

O desenvolvimento do trabalho foi baseado em atividades práticas e teóricas que auxiliaram a elaborar o modelo ilustrado pela Figura 5-1. Estas atividades estão sumarizadas na Tabela IV e envolveram o estudo de um monitor remoto, o desenvolvimento de um perfil do comportamento da rede *inf (baseline)*, o desenvolvimento de um serviço para a verificação e análise dos parâmetros de performance, o estudo de uma plataforma de gerência *(SunNet Manager)*, e finalmente, o desenvolvimento de um serviço de comunicação entre as entidades do modelo. Estas atividades estão especificadas nos próximos itens desta seção.

Tabela IV - Atividades relacionadas ao desenvolvimento do trabalho.

Componente	Atividades Desenvolvidas
agente remoto	definição do agente utilizado utilização e coleta de informações
baseline	análise dos dados obtidos utilização de um banco de dados conclusão dos resultados
serviço de verificação	cálculo de métricas de performance comparação entre os valores da *baseline* e do agente remoto constatação de tendências de degradação envio de notificações
plataforma de gerência	estudo e desenvolvimento de novas funções de gerência no *SunNet Manager* (SNM) monitoração da RMON MIB, estudo do agente *proxy* SNMP fornecido pela plataforma SNM realizações de ações de gerência
serviço de comunicação	estudo de recursos de sistemas distribuídos (*sockets)* implementação de um mecanismo de envio de notificações

5.2 Agente ou monitor remoto

Inicialmente, foi especificado um agente remoto com o auxílio da TDF LOTOS (Técnica de Descrição Formal LOTOS [ISO88]) para ser utilizado durante o desenvolvimento do trabalho [DEF95a]. No entanto, com o conhecimento do monitor de *software* (discutido na Seção 2.3.5) disponível por FTP, optou-se em utilizá-lo para

obter as informações da RMON MIB e da MIB II. Para comprovar a sua utlização no trabalho desenvolvido foram realizados alguns testes.

No RFC1271 (descrição da RMON MIB), mais especificamente no Grupo *Statistics*, existe um objeto chamado *etherDropEvents*, que identifica a perda de pacotes através da falta de recursos internos da máquina. O critério observado nestas monitorações foi a quantidade de eventos gerados por falta de recursos (pacotes perdidos pelo agente) versus o número de pacotes recebidos pela rede. Os valores da variável *etherDropEvents* observados, foram pouco significativos quando comparados ao volume de informações, em média 0,03%. Este valor é perfeitamente aceitável, considerando que em períodos de menor tráfego diminui para 0%.

5.2.1 Grupos e variáveis relevantes à performance

De acordo com a aplicação especificada na Seção 4.4 foram monitorados alguns grupos de objetos da RMON MIB (discutida na Seção 2.3.5) relevantes à performance da rede, os quais fazem parte da baseline do sistema. Além disso, também se relacionam a questões de performance, alguns objetos da MIB II [GOL93] [TAR94].

5.2.1.1 RMON MIB

Entre os grupos de objetos da RMON MIB relativos a performance que foram utilizados no modelo estão descritos na Tabela V (próxima página):

Tabela V - Grupo de objetos da RMON MIB.

Grupo	Objeto Gerenciado	Descrição
SATISTICS	*etherStatsDropEvents*	número total de eventos gerados por descartes de pacotes devido à falta de recursos
	etherPkts65to127Octets	pacotes recebidos de tamanho entre 65 e 127 octetos
	etherPkts128to255Octets	pacotes recebidos de tamanho entre128 e 255 octetos
	etherPkts256to511Octets	pacotes recebidos de tamanho entre 256 e 511 octetos
	etherPkts512to1023Octets	pacotes recebidos de tamanho entre 512 e 1023 octetos
	etherPkts1024to1518Octets	pacotes recebidos de tamanho entre 1024e 1518 octetos
HISTORY	*etherHistoryIntervalStart*	o início em que a coleta foi iniciada
	etherHistoryDropEvents	histórico do número de eventos gerados devido a falta de recursos
	etherHistoryOctets	histórico do número de bytes recebidos pela interface monitorada (inclusive pacotes com erros)
	etherHistoryPkts	histórico do número de pacotes recebidos pela interface monitorada (inclusive pacotes com erros)
	etherHistoryBroadcastPkts	histórico do número de pacotes direcionados a endereços de *broadcast*
HOSTS	*hostInPkts*	número de pacotes sem erros transmitidos para o host desde o início da monitoração
	hostOutPkts	número de pacotes (incluindo com erros) transmitidos pelo host desde o início da monitoração
	hostInOctets	número de bytes transmitidos para o host sem erros
	hostOutOctets	número de bytes transmitidos pelo host , incluindo pacotes com erros
	hostOutErrors	número de pacotes com erros transmitidos pelo hostdesde o início da monitoração
TRAFFIC MATRIX	*matrixSDSourceAddress*	endereço físico do fonte
	matrixSDDestAddress	endereço físico do destino
	matrixSDPkts	número de pacotes transmitidos entre o fonte e o destino (inclusive os erros)
	matrixSDOctets	número de bytes transmitidos, inclui todos os pacotes transmitidos
	matrixDSErrors	número de pacotes com erros transmitidos do fonte para o destino

* A variável *etherStatsDropEvents* indica o número de eventos gerados devido a falta de recursos internos do agente remoto. Este objeto indica se o agente está sendo eficiente ao coletar as informações sobre o barramento *Ethernet.*

5.2.1.2 *MIB-II*

A Tabela VI apresenta os objetos gerenciados da MIB II utilizados no modelo e relevantes para calcular as taxas de erros do segmento de rede.

Tabela VI - Grupo de objetos da MIB-II.

Grupo	Objeto Gerenciado	Descrição
Interface (IF)	*ifInDiscards*	taxa de entrada descartadas
	ifOutDiscards	taxa de transmissões descartadas
	ifInErrors	taxa de erros de entrada
	ifOutErrors	taxa de erros de saída
	ifInUcastPkts	taxa de pacotes unidirecionados recebidos
	ifOutUcastPkts	taxa de pacotes unidirecionados enviados
	ifInNUcastPkts	taxa de pacotes multidirecionados recebidos
	ifOutNUcastPkts	taxa de pacotes multidirecionados enviados
	ifOutQLen	total de pacotes na fila de saída
	ipRoutingDiscards	rotas perdidas devido a falta de espaço no *buffer*

5.2.2 Exemplos

Neste item são apresentados três exemplos utilizando dados fornecidos pelo monitor remoto instalado no ambiente de testes. O exemplo 1 apresenta o modo como podem ser interpretados os dados fornecidos pelo Grupo *Hosts* da RMON MIB. O exemplo 2 apresenta uma comparação entre o comportamento da gerência pró-ativa e da gerência reativa de redes (adotada pelos atuais sistemas de gerência). O exemplo 3 demonstra a instanciação de filtros através do Grupo *Filter* da RMON MIB.

5.2.2.1 *Exemplo 1*

O exemplo 1 apresenta o modo como são calculados os parâmetros para a avaliação de desempenho através da RMON MIB. Os valores abaixo (Tabela VII) correspondem a exatamente, 166 horas, 45 minutos e 30 segundos de monitoração (600.330 segundos).

Tabela VII - Dados da RMON MIB coletados pelo monitor.

Hosts	InOctets	OutOctets	InPkts	OutPkts	MAC Address
Vênus	1682483473	171895	10380337	2724	08:00:20:11:fd:25
Apolo	483446481	1079349867	983914	2705907	08:00:20:0d:2e:8c
Vesta	476338318	333010751	546664	562664	08:00:20:03:14:63
Atlas	63643940	431312890	342078	1384521	08:00:20:0e:9b:66
Ceres	92680989	258819603	379118	1289074	08:00:20:04:0f:f9
Mercúrio	292716788	977165353	1671600	3451756	00:00:3b:80:33:9e
Janus	1407710	43990470	5756	71436	00:00:3b:80:38:46
Grad-GW	66666232	580249263	391645	2112480	00:20:af:a4:d7:ce

Com estes dados é possível calcular a taxa de entrada e saída, a vazão (número de transações por unidade de tempo) de cada estação, em octetos ou em pacotes. Estes valores permitem ainda, calcular a freqüência com que cada estação transmite ou recebe informações na interface.

O gráfico da Figura 5-2 ilustra a distribuição da taxa de entrada dos pacotes da interface monitorada. Como pode ser observado, 64% dos pacotes recebidos são enviados à estação Vênus, 10% para a estação Mercúrio, 6% para a estação Apolo, 3% para a estação Vesta, 2% para a estação Atlas, Ceres e Grad-Gw, 1% para o endereço *Broadcast* e 9% é distribuído entre o restante dos equipamentos da sub-rede.

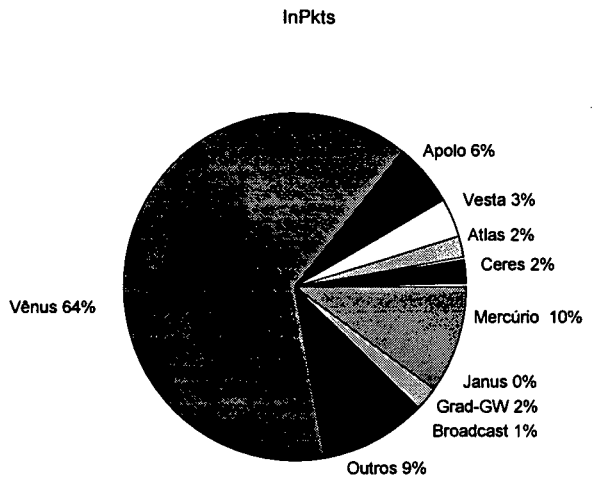


Figura 5-2 Distribuição dos pacotes recebidos pela interface.

Analogamente, a Figura 5-3 ilustra a distribuição da taxa de saída da interface. Do total de transmissões, 21% são efetuadas pela estação Mercúrio, 17% pela estação Apolo, 13% pela Grad-GW, 8% pela estação Atlas e Ceres e 3% pela estação Vesta. E cerca de 30% das transmissões são realizadas pelos demais equipamentos da interface.

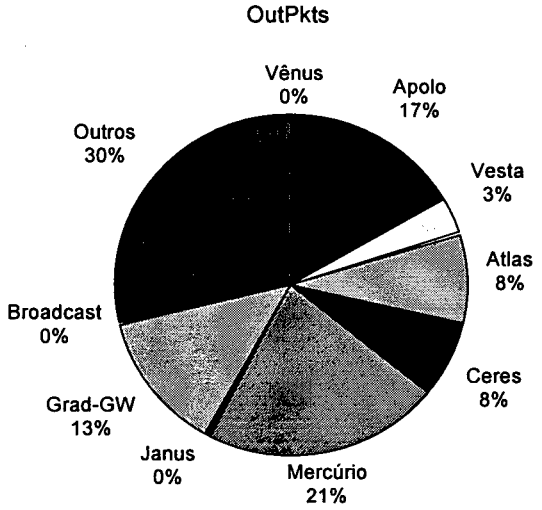


Figura 5-3 Distribuição da taxa de saída dos pacotes da interface.

5.2.2.2 Exemplo 2

O exemplo 2 apresenta uma comparação entre o comportamento reativo e pró-ativo da gerência de redes.

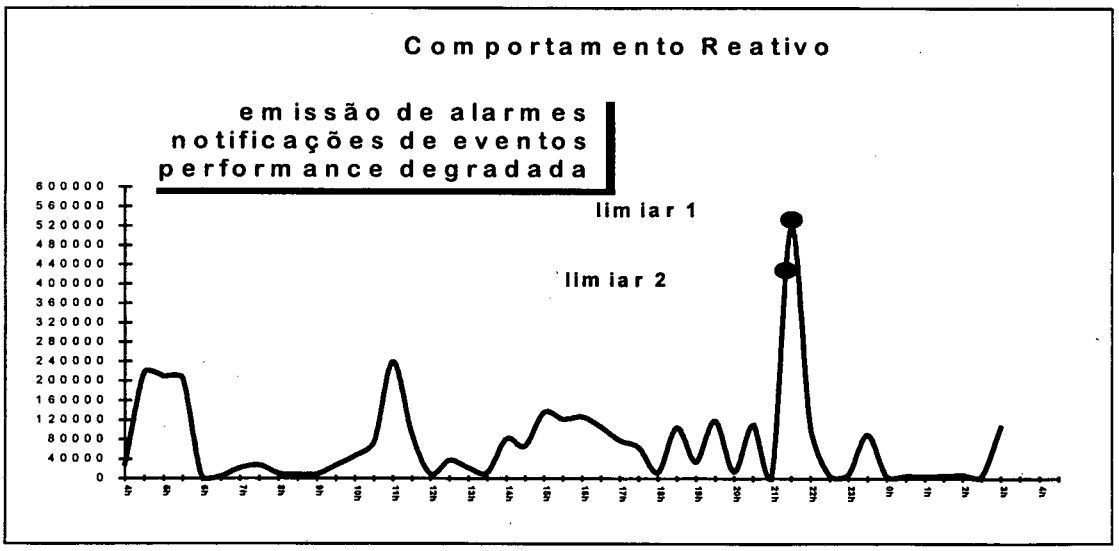


Figura 5-4 Comportamento da gerência de redes reativa.

Comportamento Reativo (filosofia de trabalho adotada pelos atuais sistemas de gerência): em um sistema de gerência reativo, normalmente são determinados limiares (*thresholds*) nos horários de maior carga (Figura 5-4), pelo gerente ou administrador da rede. Quando estes limiares são alcançados ou ultrapassados, o sistema emite alarmes ou eventos indicando o nível de tráfego alcançado. As ações para corrigir a situação são realizadas depois que a performance já está degradada. Este comportamento pode ser observado em plataformas de gerência, tais como: *SunNet Manager* da *Sun Connect*, *NetView 6000* da IBM e *OpenView* da *Hewlett Packard*.

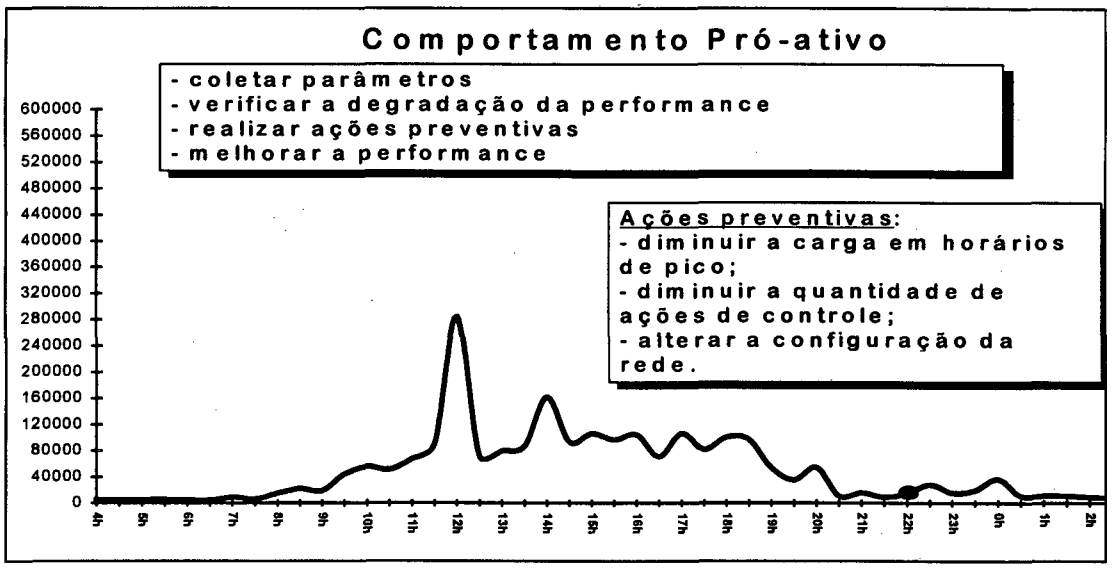


Figura 5-5 Comportamento da gerência de redes pró-ativa.

Comportamento Pró-ativo (filosofia proposta neste trabalho): em um sistema de gerência pró-ativo, com base em pesquisas sobre o comportamento da rede (*baseline*) é possível determinar ações preventivas para evitar a queda de performance em horários de carga excessiva. Deste modo seria possível minimizar o tráfego do período de pico máximo, proporcionando uma melhor utilização dos recursos da rede. Como pode ser observado pela Figura 5-5.

5.2.2.3 Exemplo 3

Os filtros são utilizados para selecionar as informações a serem monitoradas em um segmento da rede. Eles são utilizados para diminuir o fluxo de informações

monitoradas por segmento. Este recurso está disponível na RMON MIB através do Grupo *Filter*. Como o agente remoto utilizado monitora todos os pacotes que transitam na interface configurada, algumas vezes é necessária a criação de filtros para limitar o tráfego de informações. Por este motivo, foram realizados alguns testes para verificar o funcionamento do mecanismo de filtragem do monitor remoto.

Com o objetivo de selecionar os pacotes direcionados aos microcomputadores PC's da interface, foram instanciados oito filtros, dois para cada microcomputador. Quatro filtros selecionam os pacotes que possuem os endereços fontes iguais aos dos micros, e os outros quatro selecionam os pacotes com o endereço destino iguais aos dos microcomputadores. Desta forma é possível selecionar o tráfego de todos os pacotes transmitidos e recebidos pelos micros. Foram criados dois canais para onde o tráfego filtrado é direcionado, um para os pacotes enviados e outro para os pacotes recebidos pelo segmento testado. Cada um destes canais foi utilizado para capturar pacotes individuais, fornecer a tabela dos *hosts* associados a estes equipamentos e fornecer a matriz de tráfego do segmento testado.

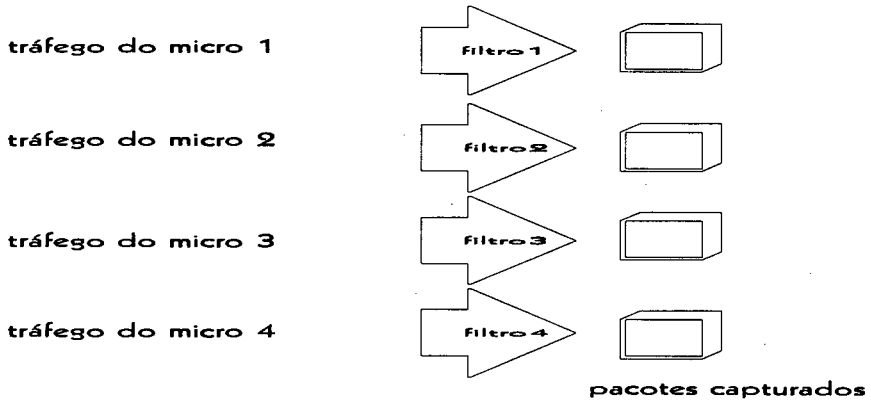


Figura 5-6 Teste realizado com microcomputadores para filtragem de informações.

Quando ao agente é solicitada uma tabela de filtros, por exemplo, o agente faz a leitura de um segmento da árvore onde as informações estão armazenadas (MIT - *Management Information Tree*) através de operações *snmp-get* e *snmp-getnext* fornecendo as informações sobre o grupo filtros e as de outros grupos que estejam relacionadas a ele. Por exemplo, em destaque está a chamada do agente para efetuar uma operação de coleta de uma tabela:

```
venus% echo "filterTable[]..."|snmp-tbl
filterIndex[10]=10; channelIndex[10]=10
filterIndex[11]=11; channelIndex[20]=20
filterIndex[12]=12; channelIfIndex[10]=1
filterIndex[13]=13; channelIfIndex[20]=1
filterChannelIndex[10]=10; channelAcceptType[20]=1
filterChannelIndex[11]=10; channelDataControl[10]=1
filterChannelIndex[12]=10; channelDataControl[20]=2
filterChannelIndex[13]=10; channelTurnOnEventIndex[10]=0
filterPktDataOffset[10]=6; channelTurnOffEventIndex[10]=0
filterPktDataOffset[11]=6; channelTurnOffEventIndex[20]=0
filterPktDataOffset[12]=6; channelEventIndex[10]=0
filterPktDataOffset[13]=6; channelEventIndex[20]=0
filterPktData[10]=08:00:00:32:01:97; channelEventStatus[20]=1
filterPktData[11]=08:00:00:32:04:13; channelMatches[10]=110350
filterPktData[12]=08:00:00:31:96:71; channelMatches[20]=10490
filterPktData[13]=08:00:00:31:98:75; channelDescription[10]=""
filterPktDataMask[10]=ff:ff:ff:ff:ff:ff; channelOwner[10]=""
filterPktDataMask[11]=ff:ff:ff:ff:ff:ff; channelOwner[20]=""
filterPktDataMask[12]=ff:ff:ff:ff:ff:ff; channelStatus[10]=1
filterPktDataMask[13]=ff:ff:ff:ff:ff:ff; channelStatus[20]=1
filterPktDataMask[20]=ff:ff:ff:ff:ff:ff; bufferControlIndex[20]=20
filterPktDataNotMask[10]=00:00:00:00:00:00;
bufferControlChannelIndex[20]=20
filterPktDataNotMask[11]=00:00:00:00:00:00; bufferControlFullStatus[20]=2
filterPktDataNotMask[12]=00:00:00:00:00:00; bufferControlFullAction[20]=1
filterPktDataNotMask[13]=00:00:00:00:00:00;
filterPktStatus[10]=0; bufferControlDownloadOffset[20]=0
filterPktStatus[11]=0; bufferControlMaxOctetsRequested[20]=-1
filterPktStatus[12]=0; bufferControlMaxOctetsGranted[20]=-1
filterPktStatus[13]=0; bufferControlCapturedPackets[20]=304
filterPktStatusMask[10]=0; bufferControlOwner[20]=""
filterPktStatusMask[11]=0; bufferControlStatus[20]=1
filterPktStatusMask[12]=0; captureBufferControlIndex[20.1]=20
filterPktStatusMask[13]=0; captureBufferControlIndex[20.2]=20
filterPktStatusNotMask[10]=0; captureBufferControlIndex[20.4]=20
filterPktStatusNotMask[11]=0; captureBufferControlIndex[20.5]=20
filterPktStatusNotMask[12]=0; captureBufferControlIndex[20.6]=20
filterPktStatusNotMask[13]=0; captureBufferControlIndex[20.7]=20
filterOwner[10]=""; captureBufferControlIndex[20.9]=20
filterOwner[11]=""; captureBufferControlIndex[20.10]=20
```

Para cada instância da tabela, é necessário determinar o que filtrar e a que canal direcionar as informações. Para isso é colocado o valor na variável *filterPktData* de cada um dos endereços *Ethernet* dos micros a serem filtrados. A variável

filterPktDataOffset determina a partir de qual posição do cabeçalho do pacote começar a leitura. E as variáveis *filterPktDataMask* e *filterPktDataNotMask* definem quais os testes a fazer com os pacotes [STA94]. Com os valores acima apresentados serão filtrados todos os pacotes que possuem os endereços fonte e destino iguais aos das placas *Ethernet* em cada micro.

5.3 Desenvolvimento da *baseline*

Uma das tarefas mais importantes, e também a mais complexa, é desenvolver a *baseline* (descrita na Seção 2.3.1). Os dados monitorados devem ser cuidadosamente analisados para que o comportamento da rede seja descrito corretamente. Para auxiliar na análise dos dados foi utilizado um banco de dados. No banco de dados foram acrescentadas as amostras obtidas com o auxílio do monitor remoto, com o objetivo de selecionar os dados que melhor representam o comportamento da rede. Depois de identificar o comportamento da rede, ele foi armazenado em um arquivo para ser consultado pelo serviço de verificação, ou passar por um simulador para identificar outros parâmetros. A Figura 5-7 ilustra as etapas realizadas para o desenvolvimento da *baseline*.

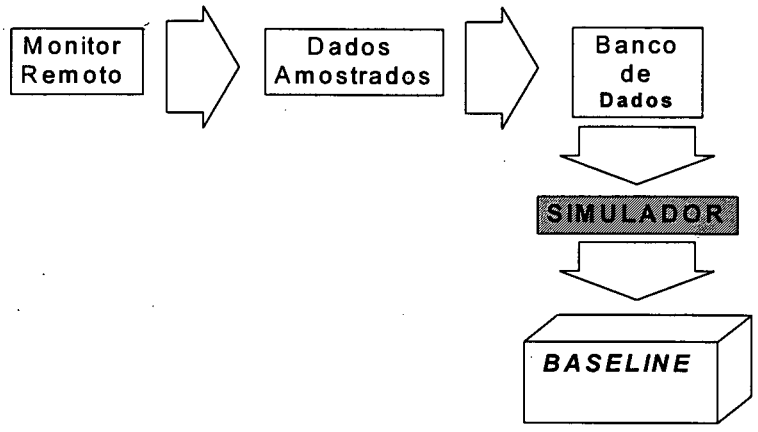


Figura 5-7 Desenvolvimento de baselines.

5.3.1 Utilização do simulador

Futuramente, serão utilizados os simuladores SIMAN V e ARENA, para identificar o comportamento da rede em novos cenários de carga. Em [BAR95] é feito um estudo para a transmissão de sinais de vídeo e áudio em um *backbone* FDDI com o auxílio destes simuladores, os quais foram utilizados para avaliar o desempenho das transmissões. O ambiente para desenvolvimento de simulações no ARENA pode ser visualizado através da Figura 5-8.

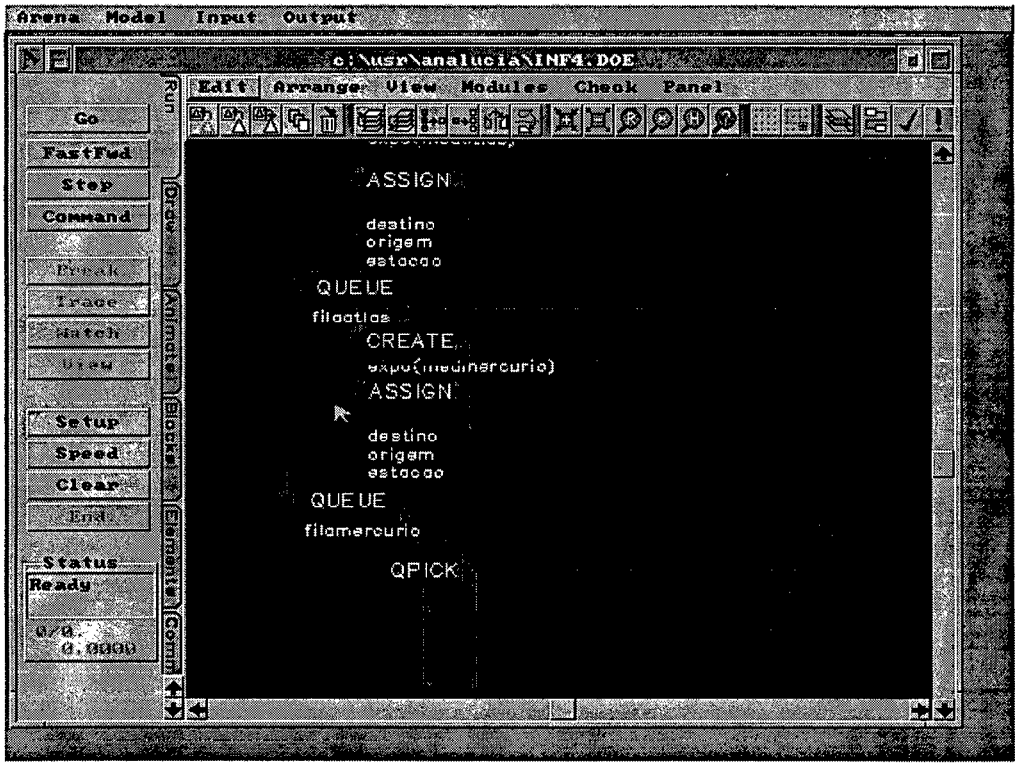


Figura 5-8 O ambiente de desenvolvimento do simulador ARENA.

Para desenvolver a *baseline* no simulador deverá ser modelado o comportamento da rede de acordo com suas características. O tipo de interface (neste caso, tipo *Ethernet*), o número de elementos, a taxa de chegada de pacotes e a distribuição do tamanho dos pacotes são exemplos de características que deverão ser consideradas durante o desenvolvimento do modelo a ser simulado. Os resultados das monitorações da rede devem ser utilizados neste modelo para garantir a integridade das informações simuladas.

Outro fator importante, é que a simulação pode ser utilizada para verificar alguns aspectos difíceis de serem determinados apenas com o uso de monitorações, como a capacidade máxima da rede, por exemplo. Deste modo, pode-se desenvolver *baselines* mais confiáveis, armazenando-se todas informações relativas à área da aplicação em desenvolvimento.

5.3.2 Métricas de performance

Dentre as métricas de performance que devem compor uma *baseline,* destacam-se as relacionadas na Tabela VIII:

Tabela VIII - Métricas de performance da *Baseline.*

Métricas	Descrição
Utilização da linha	é a razão entre o total de octetos por segundo e a capacidade máxima da linha de transmissão
Taxa de colisões	é o número de colisões por pacote transmitido
Taxa de entrada	é o volume de informações recebidas pela linha
Taxa de saída	é o volume de informações transmitidas pela linha
Vazão	é o número de transações por unidade de tempo
Freqüência de utilização das estações	é a freqüência com que cada estação transmite ou recebe informações pela linha de comunicação

5.4 Serviço de verificação

Este serviço compara os dados reais fornecidos pelo agente remoto com os dados armazenados na *Baseline.* Ele deve identificar os tipos de degradação quando ocorrerem, e emitir notificações ao sistema de gerência. A comunicação deve ser realizada através de um serviço de comunicação. O sistema de gerência e o administrador da rede devem realizar ações de gerência para evitar os problemas proativamente.

Supondo, por exemplo, que o gráfico da Figura 5-9 demonstre o comportamento observado entre *vazão* e tempo.

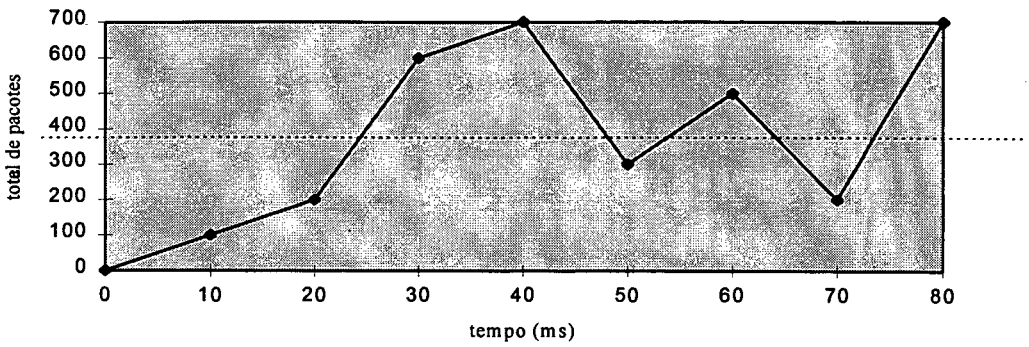


Figura 5-9 Gráfico exemplo de baseline.

E que a linha pontilhada representa a média de pacotes de entrada e saída na rede, é possível verificar que os pontos abaixo desta linha significam um desvio da qualidade do serviço oferecido pela rede. Esta linha pode representar um *threshold* (ou limiar), por exemplo. Imagine os valores do gráfico como uma *baseline* representando o comportamento da rede. Então, o serviço de verificação considera estes valores e compara-os com os resultados do agente remoto, emitindo uma notificação para a plataforma de gerência através do mecanismo de comunicação entre entidades.

5.5 Serviço de Comunicação

Apesar do agente remoto utilizar o protocolo SNMP (*Simple Network Management Protocol*) e o sistema de gerência SNM (*SunNet Manager* que é utilizado para validar a gerência pró-ativa) também, é necessário um agente que permita a troca de informações entre o sistema de gerência e o agente remoto. Porque as variáveis do sistema SNM não são compatíveis com as variáveis da RMON MIB. Por este motivo, foi desenvolvido um mecanismo que permite a comunicação entre o monitor remoto e o agente *proxy* do SNM. Um agente *proxy* tem como principal objetivo fazer a comunicação entre um agente e os objetos gerenciados que não fazem parte do mesmo protocolo.

Existem dois meios para construir o serviço de comunicação entre as entidades do sistema. O primeiro é apresentado baseado na utilização de *sockets*, e o outro é baseado na construção de um subagente para agentes *proxies* SNMP.

5.5.1 Comunicação através de sockets

Uma das maneiras de construir um serviço de comunicação entre as principais entidades do sistema é utilizando recursos conhecidos como *sockets*. Através destes recursos é possível criar um mecanismo de comunicação utilizando um protocolo da camada de transporte (TCP ou UDP, Seção 2.2) da arquitetura Internet.

5.5.1.1 Sockets

Um bloco básico para a comunicação entre processos é chamado *socket*() [SUN90].Um *soket* é um ponto final de uma comunicação. Cada um possui um tipo e um ou mais processos associados.

5.5.1.2 Tipo de sockets

A comunicação deverá ser realizada entre tipos de *sockets* que determinam qual o tipo de comunicação que será realizada. Os tipos de *sockets* disponíveis pelo SunOS [SUN90]:

- *stream*: fornece fluxo de dados bidirecional, confiável, seqüenciado, sem duplicação e sem registro de limitações.
- *datagram*: suporta fluxo de dados bidirecional porém sem garantias de confiabilidade, seqüencialidade e não duplicidade. Por exemplo, um processo que recebe uma mensagem sobre um *socket* datagrama pode encontrar mensagens duplicadas, e possivelmente ordem em que foi enviada.
- *raw*: fornece acesso aos usuários para protocolos da comunicação de baixo nível ou interface de redes.

5.5.1.3 Utilização

A maneira de realizar a comunicação entre duas entidades, em um ambiente UNIX, é através do mecanismo de *sockets* e um protocolo de comunicação. Neste trabalho foi criado uma mensagem específica que atuará como uma notificação

(operação *trap* do SNMP) para a gerência pró-ativa. Ambos os protocolos de transporte podem ser utilizados, TCP (serviço confiável) ou UDP (serviço não confiável) [SUN90]. Para criar um *socket* de comunicação entre a plataforma de gerência e o módulo de gerência pró-ativa (localizados em domínios diferentes) utilizando datagramas, faz-se por exemplo:

s = socket(AF_INET, SOCK_DGRAM, 0);

A comunicação entre os processos é realizada através de uma associação. Na Internet uma associação é composta por um de um endereço local e outro remoto, portas locais e remotas. Uma associação deve ser única.

Uma associação *Internet* é formada pela seguinte tupla:

<protocolo, endereço local, porta local, endereço remoto, porta remota>

A mensagem *trap* para notificar a entidade de gerência sobre as condições anormais no segmento de rede, deve conter as informações ilustradas pela Figura 5-10.

Figura 5-10 Formato da mensagem enviada para a plataforma de gerência.

porta destino: endereço da estação de gerência responsável pelo segmento;

porta fonte: endereço do segmento gerenciado que provavelmente terá problemas de performance;

hosts: endereço físico dos *hosts* que estão degradando a performance dentro do segmento;

parâmetros: os parâmetros da MIB que estão demonstrando degradação, para que o gerente possa modificá-los antes de perder a performance;

reservado: espaço reservado para aperfeiçoamentos futuros.

5.5.2 Comunicação através de um subagente proxy

Outra forma de realizar a comunicação entre entidades do sistema que foi estudada é através de um subagente *proxy* SNMP [ANG95][QUI95].

A maioria das plataformas de gerência possui um agente *proxy* SNMP, normalmente chamado **snmpd** (SNMP *daemon*). Este agente permite a troca de informações entre entidades SNMP e os sistemas de gerência. A construção de um subagente que utilize os recursos já fornecidos pela plataforma permite que o sistema de gerência reconheça os objetos gerenciados pelo monitor remoto. Desta forma, é possível a criação de um *trap* específico que seja reconhecido pelo sistema de gerência através do subagente.

A criação do subagente fornece várias vantagens sobre a utilização de *sockets*, entre elas:

- a programação através de API's (*Application Programming Interfaces*), as bibliotecas de serviços que normalmente acompanham as plataformas de gerência permitem uma programação de sistemas de comunicação de forma mais legível e estruturada;
- a utilização de *sockets* exige maior conhecimento de sistemas distribuídos e programação em redes;
- com o uso de APIs não é necessário criar uma interface com o usuário, porque ela permite o interfaceamento direto com a plataforma de gerência.

Capítulo 6

Implementação do Protótipo

6. Aspectos de Implementação

Este capítulo apresenta os aspectos de implementação do desenvolvimento do protótipo experimental para viabilizar a gerência pró-ativa de redes.

6.1 Protótipo

Com o objetivo de validar a gerência pró-ativa proposta neste trabalho foi desenvolvido um protótipo, o qual possui algumas características básicas para o desenvolvimento de uma aplicação para avaliação de desempenho pró-ativa de redes. Os componentes de *hardware* e *software* que compõem o protótipo estão descritos a seguir.

6.1.1 Componentes de *hardware*

As estações da sub-rede *inf* que fazem parte do protótipo implementado estão descritas na Tabela IX. Estas estações representam cerca de 85% do tráfego da sub-rede. Para simplificar, foram descartados os demais componentes (aproximadamente 40 elementos) que representam menos de 20% das informações que transitam na sub-rede.

Tabela IX - Estações da sub-rede *inf.ufsc* que fazem parte do protótipo.

Nome (endereço IP)	Especificação	Sistema Operacional	Localização
vênus (150.162.60.1) Mail Exchanger e Gateway	Sparc 10, CPU SPARCStation 10, 32 MB, 1.7 GB disco, 01 unidade de CD-ROM Sun	SunOS 4.1.3.	Lab PGCC
apolo (150.162.60.2)	Sparc IPX, CPU SPARCStation IPX, 16 MB,	SunOS 4.1.3.	Lab PGCC
vesta (150.162.60.3)	Sparc SLC (diskless), 8 MB	SunOS 4.1.3.	LISHA
atlas (150.162.60.4)	Sparc 2, 16 MB, 800 MB disco	SunOS 4.1.3.	Labine
ceres (150.162.60.5)	Sparc ELC, 8MB, 424 MB disco	SunOS 4.1.3	Lab PGCC
mercurio(150.162.60.9)	Axill 240, 48 MB, 2GB disco	SunOS4.1.3_ U1	LabPGCC
janus (150.162.60.10)	Axil 220, 48 MB, 2GB disco	SunOS .1.3_U1	Labine
grad-gw(150.162.60.100)	486 DX, 12 MB, 670 MB disco	SCO UNIX 3.2	Labine

A função de estação de gerência do protótipo é desempenhada pela estação Orion, localizada na sub-rede *lrg.ufsc.br* (o ambiente de testes é discutido na Seção 4.2). A Orion é uma *SunSparc 20,* na qual está instalada a plataforma de gerência *SunNet Manager versão 2.2* (SNMv.2.2) e o sistema operacional *Solaris 2.4* (como descrito no ambiente de testes, Seção 4.1).

6.1.2 Componentes de *software*

Para o desenvolvimento do protótipo de gerência pró-ativa foram utilizados um agente SNMP, a plataforma de gerência SNMv.2.2, um banco de dados e um compilador da linguagem C.

O agente remoto foi utilizado para coletar as informações da RMON MIB sobre as estações monitoradas (discutido na Seção 2.4.4). Foi instalado na estação Vênus, servidora da sub-rede *inf.ufsc.br*, e está descrito na Seção 2.3.4. Com exceção da plataforma SNM e do banco de dados, os demais componentes do protótipo foram desenvolvidos utilizando a linguagem C (o filtro, o serviço para a verificação e o envio de notificações).

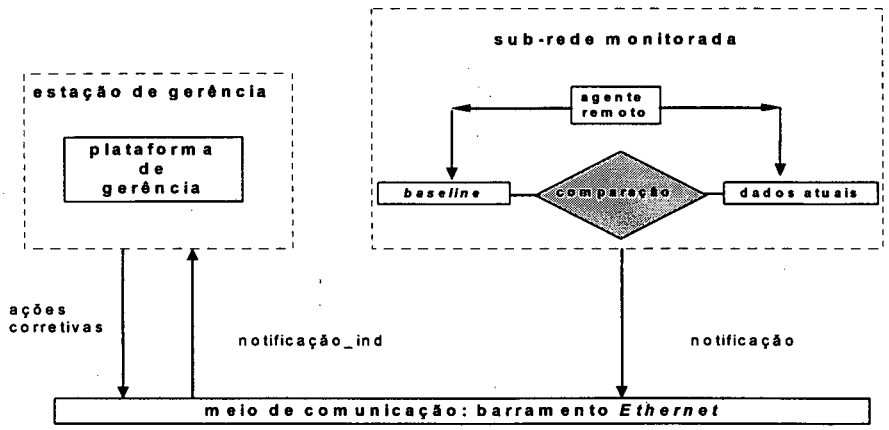


Figura 6-1 Visão geral do protótipo.

A Figura 6-1 apresenta as duas sub-redes interligadas através do meio de comunicação (neste caso, um barramento *Ethernet*). As notificações são enviadas através do meio para a estação responsável pelo gerenciamento da sub-rede monitorada. Esta estação deve realizar as ações corretivas para evitar a degradação da performance.

6.2 Implementação

Nesta seção estão descritos os módulos desenvolvidos em linguagem C e a construção da *baseline* com o auxílio de um banco de dados.

6.2.1 Serviço de verificação

O serviço de verificação foi desenvolvido em linguagem C, com o uso de recursos para a programação em redes (*sockets*) fornecidos pelo sistema operacional *SunOS* [SUN90]. Os componentes do protótipo foram desenvolvidos com base no modelo proposto no Capítulo 5, Seção 5.1. Este serviço é composto por três módulos básicos, os quais podem ser visualizados pelo diagrama da Figura 6-2, e estão descritos abaixo:

Módulo I - faz a leitura, a filtragem e o cáculo dos parâmetros de desempenho, a partir dos dados fornecidos pelo agente remoto;

Módulo II - faz a leitura dos dados da *baseline* e compara-os com os resultados do Módulo I;

Módulo III - envia notificações para a estação de gerência da rede.

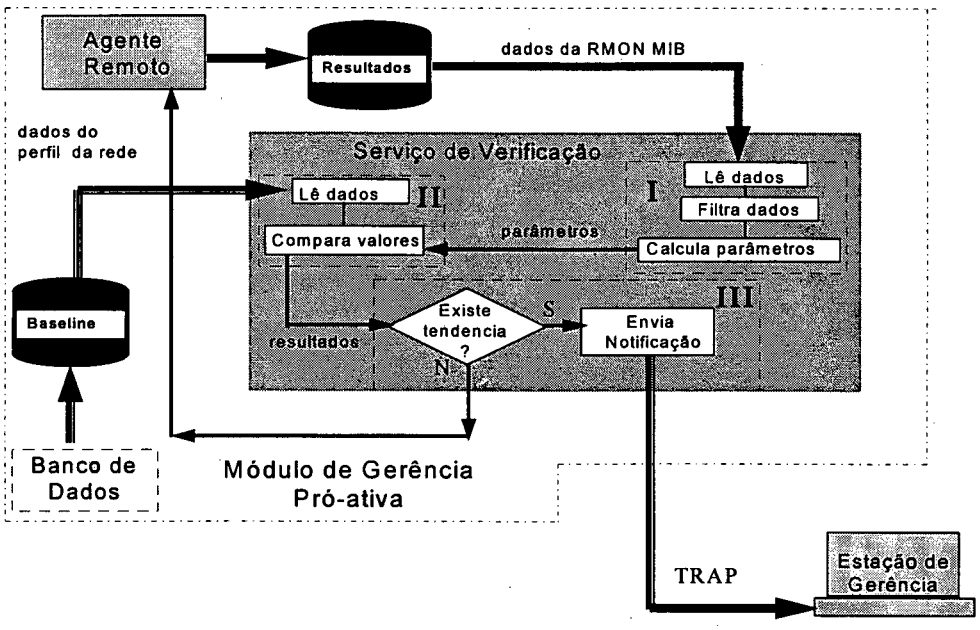


Figura 6-2 Módulos do protótipo implementados.

6.2.1.1 Módulo I

O objetivo deste módulo no serviço de verificação é coletar informações provindas do agente remoto, filtrar as informações relacionadas às estações do protótipo e calcular os parâmetros de avaliação de desempenho. Este módulo possui duas funções principais, uma que calcula o somatório das variáveis das estações do protótipo e outra que calcula a utilização de cada estação. O código fonte deste módulo é apresentado no Anexo II deste documento. A Figura 6-3 apresenta o relacionamento entre as funções que compõem este módulo.

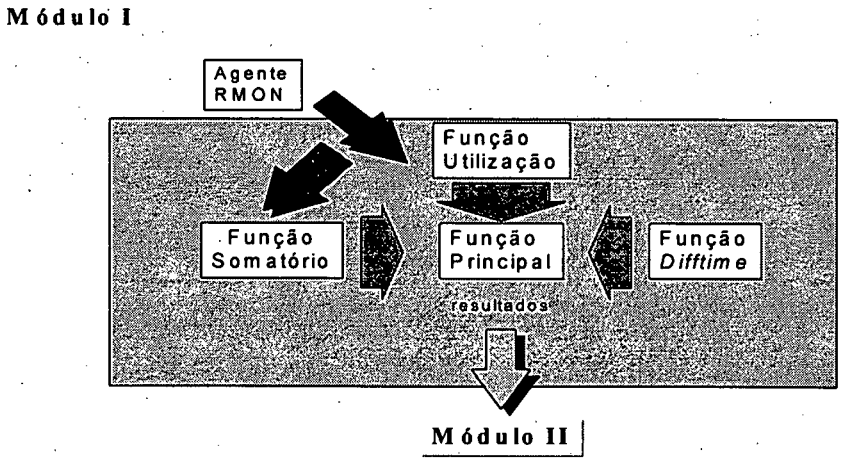


Figura 6-3 Relacionamento das funções que compõem o Módulo I.

A **função somatório** recebe como parâmetros um nome de arquivo e o tipo de variável que vai ser somada. Os tipos de variáveis aceitas fazem parte de um subconjunto de objetos da RMON MIB. Nesta função é realizada também a filtragem das informações, na qual somente são consideradas e somadas as variáveis relativas ao subconjunto de estações que fazem parte do protótipo.

A **função utilização** recebe os mesmos parâmetros da função somatório, descrita acima. No entanto, sua principal função é relacionar a frequência com que cada estação recebe ou transmite dados, caracterizando uma espécie de utilização.

Como a versão do compilador utilizada não possuía a **função *difftime***, a mesma foi desenvolvida para calcular a diferença de tempo entre dois intervalos de

coleta, um x e outro y, em segundos. Recebe como parâmetros, um tempo inicial e outro final do tipo *time_t* definido pela biblioteca *time.h* da linguagem C.

Na **função principal** deste módulo são calculados os parâmetros relacionados ao desempenho da rede. Entre eles:

- **Taxa de entrada em pacotes**: para calculá-la é obtido o horário da coleta (x) e o somatório de pacotes recebidos (objeto *hostTimeInPkts* da RMON MIB); após alguns milisegundos é coletada uma nôva amostra no tempo y. Por último, aplica-se:

$$inputRatePkts = \frac{Pacotes\_recebidos_y - Pacotes\_recebidos_x}{y - x} \quad (6\text{-}1)$$

- **Taxa de saída em pacotes**: para calculá-la é obtido o horário da coleta (x) e o somatório de pacotes transmitidos (objeto *hostTimeOutPkts* da RMON MIB, após alguns milisegundos é coletada uma nova amostra em tempo y. Por último, aplica-se:

$$outRatePkts = \frac{Pacotes\_transmitidos_y - Pacotes\_transmitidos_x}{y - x} \quad (6\text{-}2)$$

- **Taxa de entrada em *bytes***: para calculá-la é obtido o horário da coleta (x) e o somatório do número de *bytes* recebidos (objeto *hostTimeInOctets* da RMON MIB); após alguns milisegundos é coletada uma nova amostra em tempo y. Por último, aplica-se a mesma fórmula da taxa de entrada em pacotes (*inRatePkts* descrita acima), porém para o número de *bytes* recebidos.

- **Taxa de saída em *bytes***: para calculá-la é obtido o horário da coleta (x) e o somatório do número de *bytes* transmitidos (objeto *hostTimeOutOctets* da RMON MIB); após alguns milisegundos é coletada uma nova amostra em tempo y. Por último, aplica-se a mesma fórmula da taxa de saída em pacotes (*outRatePkts* descrita acima), porém para o total de *bytes* transmitidos.

- **Vazão (ou *throughput*) em pacotes por segundo**:

$$Throughput = inRatePkts + outRatePkts \quad (6\text{-}3)$$

- **Vazão em *bytes* por segundo**:

$$Throughput = inRateBytes + outRateBytes \quad (6\text{-}4)$$

- **Utilização da linha**: Largura de banda que está sendo utilizada pela capacidade máxima da rede.

$$Utilization = \frac{ThroughputBytes * 8}{Max\_Capacity} \quad (6\text{-}5)$$

Neste trabalho (*Ethernet a 10Mbits*):

$$Utilization = \frac{ThroughputBytes * 8}{10Mbps}$$

- *Taxa de Colisões* : número de colisões sobre o total de pacotes transmitidos.

$$CollissionsRate = \frac{Collisions\_total}{outputRatePkts} \quad (6\text{-}6)$$

6.2.1.2 Módulo II

Este módulo tem como objetivo ler as informações recebidas pelo Módulo I e compará-los com a *baseline*. Se o valor da eficiência da rede armazenado na *baseline*, por exemplo, for entre 65 e 80% para até 4 estações prontas para transmitir, e entre 40 e 65% para casos acima de 8 estações. Se o Módulo I repassar a seguinte informação: 40% com 10 estações prontas para transmitir. O Módulo II irá comparar com a *baseline* e verificará uma tendência de degradação da performance da rede com relação a eficiência. Neste caso, será repassado ao Módulo III um argumento que identifique tal situação. A Figura 4.6 ilustra o relacionamento entre as funções do Módulo II.

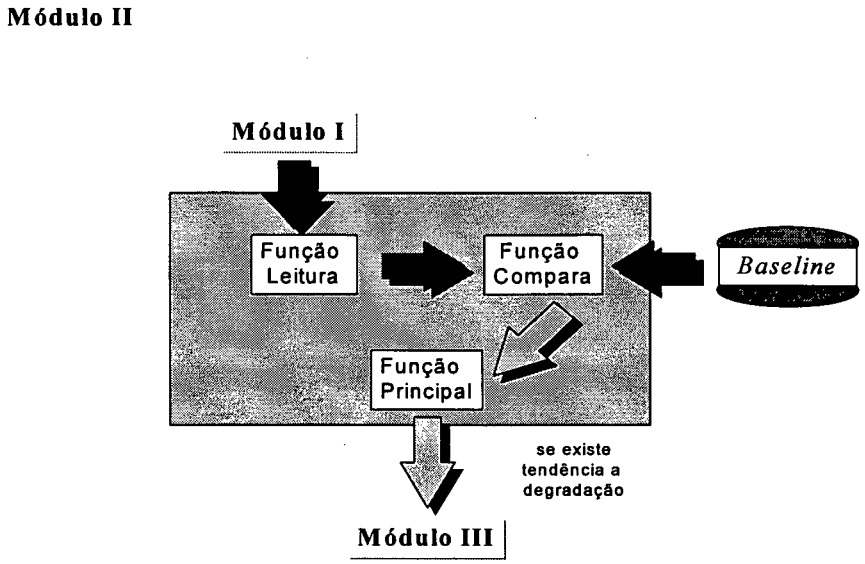


Figura 6-4 Relacionamento das funções que compõem o Módulo II.

6.2.1.3 Módulo III

Este módulo tem como principal objetivo emitir um *trap* (operação de notificação do SNMP) a estação responsável pelo gerenciamento do protótipo (a estação Orion). No entanto, esta notificação é diferente das notificações definidas pelo protocolo SNMP. Ela foi elaborada para avisar o gerente, de uma forma mais inteligente, sobre o que está acontecendo com a sub-rede e não apenas emitir um som, como os sistemas de gerência normalmente fazem. O gerente recebe uma mensagem contendo o nome da estação, a localização e um aviso de tendência para a degradação do sistema. As ações corretivas devem ser emitidas pelo gerente da rede, como por exemplo, a verificação do cabeamento e o particionamento da rede. A Figura 6-5 apresenta o relacionamento entre as funções do Módulo III.

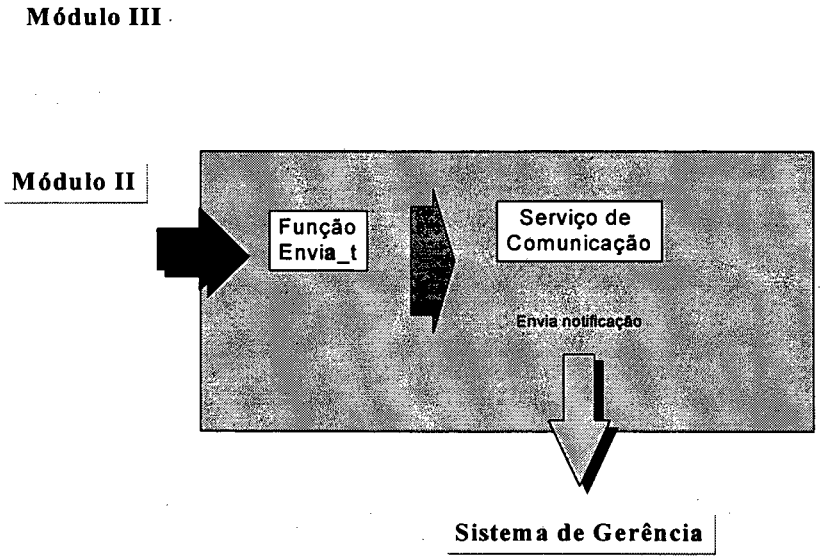


Figura 6-5 Relacionamento das funções que compõem o Módulo III.

6.3 Geração da *baseline*

A *baseline* foi desenvolvida com o auxílio de um banco de dados para analisar de forma mais eficiente os dados monitorados. Futuramente, pretende-se desenvolver uma metodologia para o desenvolvimento de *baselines*, devido à complexidade de análise das informações.

6.3.1 Utilização do banco de dados

Para facilitar a análise dos dados coletados foi utilizado um banco de dados. Os valores analisados indicarão o comportamento que melhor se adapta à rede. Esta análise ajuda a minimizar a probabilidade de erros na formação do perfil (ou *baseline*) da rede, uma espécie de filtragem das informações amostradas.

Com o auxílio do banco de dados foram analisadas as informações coletadas pelo monitor remoto: dados estatísticos da interface (Grupo *Statistics)*, histórico da interface monitorada (Grupo *History)*, taxas de entrada e saída das estações da interface, vazão (Grupo *Hosts)*, e dados sobre o tráfego entre as estações (Grupo *Traffic Matrix*).

6.3.1.1 Grupo Statistics

A partir das amostras coletadas, observou-se uma grande incidência de pacotes pequenos trafegando na sub-rede analisada, cerca de 65,7% (ver Tabela X). Isto significa que 90% do tráfego de pacotes que circula na sub-rede possui tamanho relativamente pequeno, na faixa de 64 a 255 octetos, e apenas 10% possui tamanho entre 256 e 1518 octetos.

Além disso, a eficiência da linha varia entre 50 e 80% (para até 4 estações) e 30 a 65% para um número maior de estações interconectadas à interface. Enquanto que, para pacotes entre 256 e 1518 octetos, a eficiência varia entre 80 a 95% (para até 4 estações) e entre 65 a 88% para mais de 8 estações.

Tabela X - Distribuição do tamanho de pacotes na sub-rede analisada.

Eficiência da linha		
minima: 30%	média: 65%	máxima: 88%
Tamanho dos pacotes	Distribuição	Eficiência
Até 64 Octetos	10.2%	30%
De 65 a 127 Octetos	14.1%	45%
De 128 a 255 Octetos	65.7%	65%
De 256 a 511 Octetos	1.4%	75%
De 512 a 1023 Octetos	2.8%	80%
De 1024 a 1518 Octetos	5.8%	88%

Conclui-se, portanto, que de cada 10 pacotes que circulam na rede, 9 são da faixa entre 64 e 255 octetos, e apenas 1 pertence a faixa entre 256 e 1518 octetos. O que

indica que a eficiência da rede pode variar entre 30 e 65% (confirmando o estudo de Metcalfe & Boggs, discutido na Seção 4.2.2). Então na *baseline* foi armazenado o conteúdo da Tabela X.

6.3.1.2 Grupo History

O agente remoto registra amostras periódicas dos objetos do Grupo *History* (RMON MIB RFCs 1271 e 1757), construindo um histórico com as estatísticas da interface monitorada. São realizados dois períodos de *polling*, um longo (30 em 30min) e outro curto (de 30 em 30seg).

Este histórico permite estimar as taxas de utilização e de colisões durante 24 horas de monitoração. A utilização é a quantidade de informações (*em bits*) transmitidas por segundo sobre a capacidade máxima (ou velocidade da interface), que neste caso é de 10Mbits/seg. Enquanto que a taxa de colisões é calculada a partir do total de colisões sobre o total de pacotes transmitidos durante o intervalo de amostragem.

O seguinte comportamento foi observado:

- A interface apresenta períodos com uma utilização mínima intercalados por períodos ociosos, normalmente no período compreendido entre 0h e 8h (observação considerando todas as amostras analisadas);
- O pico máximo de utilização (da faixa de 21%) cerca de 21% da capacidade de transmissão da linha foi ocupado durante o intervalo das 20h30min às 21h (ver o primeiro gráficos no Anexo III). Nas demais amostras, observou-se valores entre 12 a 18.6% em faixas de horários distintos;
- Valores médios de utilização nos demais períodos: 6%;
- A taxa máxima de colisões observada, em todas as amostras, foi de 1%. Isto significa que 1 pacote foi envolvido em colisão dentre 100 pacotes transmitidos de forma bem sucedida, uma taxa aceitável.

No entanto, existem alguns casos que ainda estão sendo analisados e que foram considerados relevantes para o comportamento de uma sub-rede. Como por exemplo, observe que após 20 horas do início da monitoração (ver gráficos do Anexo

de 200.000 pacotes), os demais pacotes que entram na interface são distribuídos entre os equipamentos restantes da sub-rede *inf*;

b.**Taxa de Saída**: a maior taxa de transmissão, em pacotes, pertence a estação Mercúrio (cerca de 1.500.000 pacotes), seguida da estação Apolo (cerca de 1.000.000 pacotes), Ceres (cerca de 700.000 pacotes), Grad-Gw (cerca de 600.000 pacotes) e Atlas (com 500.000 pacotes transmitidos). A estação Vênus é uma das estações com a taxa de saída mais baixa.

A nível de octetos:

a.**Taxa de Entrada**: a estação Vênus apresentou uma taxa de 650.000.000 de octetos, seguida da estação Mercúrio (cerca de 100.000.000 de octetos), a Ceres e a Apolo cerca de 50.000.000 de octetos e a Grad-gw e Atlas apresentaram um número inferior a este;

b.**Taxa de Saída:** a estação Mercúrio apresentou uma taxa de 380.000.000 octetos, seguida da estação Apolo (200.000.000 de octetos), a Ceres e Grad-gw (cerca de 150.000.000 de octetos) e a Atlas com a taxa de 120.000.000 de octetos. O valor restante em octetos é transmitido pelas demais estações ligadas à interface.

Tabela XII - Distribuição do tráfego da sub-rede monitorada.

Baseline			
Estação	%Octetos Recebidos	%Octetos Transmitidos	% Trans. e Recebe
Vênus	63%	0%	63%
Mercúrio	7%	50%	57%
Apolo	6%	11%	17%
Ceres	4%	7%	11%
Atlas	3%	7%	10%
Grad-gw	1.8%	7.6%	9.4%
Janus	1%	2%	3%
Vesta	1%	1.2%	2.2%
Outros equipamentos	13.2%	18.3%	31.5%

A partir das amostras observadas foram estimados os valores da distribuição do tráfego na sub-rede monitorada. As informações descritas na Tabela XII foram incluídas no arquivo da *baseline*. A distribuição dos Octetos Transmitidos indica que a estação Mercúrio garantiu o acesso ao meio em 50% das vezes em que foi

III), a taxa de utilização da linha é de aproximadamente 6% e a taxa de colisões é de 1% (dados coletados no período: 31 colisões em 3176 pacotes transmitidos).

A partir desta observação, os dados que farão parte da *baseline* são mostrados abaixo (Tabela XI):

Tabela XI - Dados da *baseline* sobre a utilização da linha e a taxa de colisões.

	Baseline		
	Mínima	Média	Máxima
Utilização da Linha	0%	7%	21%
Taxa de Colisões	0%	0.05%	1%

A taxa de colisões de 10% pode ser considerada preocupante (1 pacote colide a cada 10 transmitido) e acima de 30% é considerado agravante (3 pacotes colidem a cada 10 transmitidos). De acordo com o relato em [BOG 88], a taxa de utilização relaciona-se diretamente com o tamanho dos pacotes. Foi comprovado que pacotes menores implicam, normalmente, em baixas taxas de utilização. Enquanto que em interfaces com o tráfego de pacotes maiores, implicam em utilizações altas. Portanto, de acordo com as análises descritas no item anterior (discutidas na Seção 6.3.1.1) (para 9 pacotes menores - entre 64 a 255 octetos - que circulam na rede, trafega 1 maior - entre 256 a 1518 octetos) a taxa média de utilização para a rede analisada, com excessão dos picos máximos, confirma o comportamento descrito em tal relatório.

6.3.1.3 Grupo Hosts

O grupo de objetos *Hosts* da RMON MIB permite estimar as taxas de entrada, as taxas de saída e a vazão de cada estação do segmento monitorado. A partir dos valores amostrados foi possível determinar quais as estações possuem as maiores taxas de entrada e saída (ver gráficos do Anexo III).

A nível de pacotes:

a.**Taxa de Entrada**: a estação que possui a maior taxa de entrada é a estação Vênus (cerca de 3.500.000 pacotes), seguida pela estação Mercúrio (cerca de 700.000 pacotes), Ceres e Apolo (menos de 500.000 pacotes), Atlas e Grad-GW (menos

transmitida alguma informação, 11% a estação Apolo, 7.6% a Grad-GW, 7% a Ceres e a Atlas, 2% a Janus, 1.2% a Vesta e 18.3% foi transmitido pelos demais equipamentos da sub-rede.

A partir dos valores fornecidos pelo Grupo *Hosts,* também foi possível determinar a vazão de cada estação do protótipo listados na Tabela XIII.

Tabela XIII - Vazão (ou *throughput*) das estações do protótipo.

Baseline						
Nome da Estação	Octetos/Segundo			Bits/Segundo		
	Mínimo	Médio	Máximo	Mínimo	Médio	Máximo
Vênus	36700.8	86744.0	118926.44	293606.46	693952.0	951411.52
Mercúrio	22721.36	67215.15	101489.64	181770.88	537721.50	811917.12
Apolo	11267.61	18075.93	24906.52	90140.88	144607.44	199252.16
Ceres	8363.35	12664.72	16946.76	66906.8	101317.76	135574.08
Atlas	6328.39	11035.37	16912.25	50627.12	88282.96	135298.0
Grad-Gw	6282.31	10414.87	15138.0	50258.48	83318.96	121104.0
Janus	3618.0	3659.89	5819.29	28944.0	29279.12	46554.32
Vesta	2884.31	3336.05	3669.40	23074.48	26688.4	29355.20

O gráfico da Figura apresenta a vazão das estações do ambiente de testes do protótipo. Como estas estações representam cerca de 85% do tráfego da sub-rede, os demais componentes não foram contabilizados.

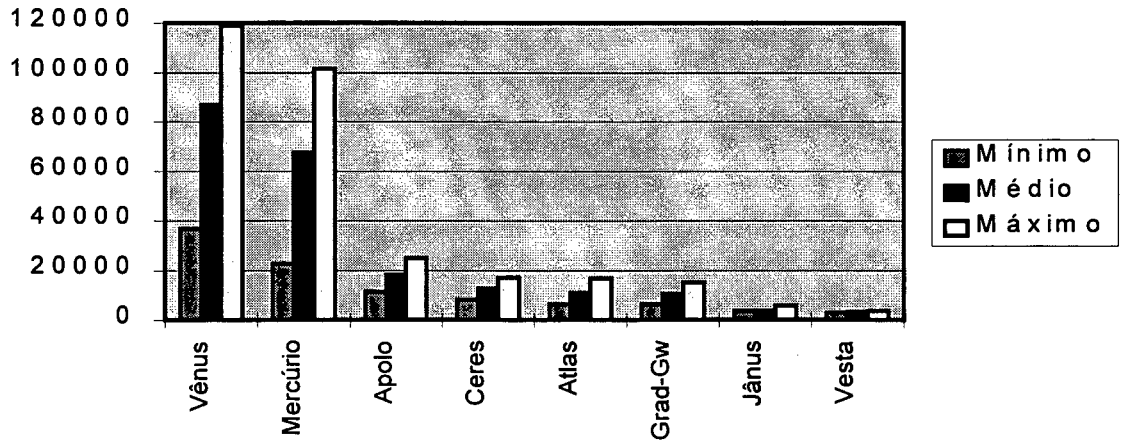


Figura 6-6 Vazão das estações do protótipo.

6.3.1.4 *Grupo Traffic Matrix*

Com base no comportamento observado na Seção 6.3.1.3, foram coletadas amostras sobre a matriz de tráfego fornecida pelo grupo Traffic Matrix. O objetivo de analisar esta matriz é verificar para onde são enviadas as informações que chegam na estação Vênus. Esta estação é responsável pelo recebimento das mensagens de correio eletrônico da sub-rede, e atua como servidora e roteadora (ver Gráficos no Anexo III).

A análise interna do tráfego da sub-rede não será incluída nesta *Baseline,* no entanto será desenvolvida, futuramente, uma nova aplicação para este protótipo. Pretende-se desenvolver um controle de tráfego para a gerência pró-ativa.

6.4 Serviço de comunicação

Apesar da plataforma SNM fornecer um agente *proxy* chamado **snmpd** (SNMP *Daemon*) que permite a troca de informações entre entidades SNMP e SNM [SUN89][SUN89a][SUN93][SUN93a][ANG95][DEF95a], não foi possível utilizar o monitor remoto com o SNM. Foi impossível monitorar a sub-rede através da plataforma devido a problemas de configuração que ainda estão sendo testados (provavelmente, entre os sistemas operacionais Solaris 1.x e Solaris 2.x). Pretende-se instalar o agente remoto em outra máquina com o sistema operacional Solaris 2.x, para utilizar as vantagens oferecidas pelo SNM (discutido na Seção 5.2.2).

Portanto, desenvolveu-se com o auxílio de *sockets* um mecanismo para transmitir notificações para a estação de gerência (Orion, ver ambiente de testes, Seção 4.1).

6.4.1 Comunicação através de sockets

A implementação foi realizada em linguagem C e com o auxílio dos *sockets* foram desenvolvidos dois programas, um cliente e outro servidor.

O programa servidor (*trapd*) oferece o serviço de comunicação na porta 1488, na qual fica aguardando uma solicitação para o envio de uma mensagem do

programa cliente. O trecho de programa abaixo ilustra a criação do *socket* do programa servidor.

```
/*************************************************************************/
/*  Constroi o nome  do socket correspondente ao canal de solicitacao de  */
/*  servicos ao servidor tty.                                             */
/*************************************************************************/

        bcopy(hp->h_addr, &name_serv.sin_addr, hp->h_length);
      name_serv.sin_family = AF_INET;
      name_serv.sin_port = htons(1488);

      if (sendto(sock,(char *)msg,sizeof msg,0,(struct sockaddr*)&name_serv,sizeof
(name_serv)) < 0)
            {
             perror("Error - sending datagram message");

             exit(0);
          }
```

Quando ocorre uma notificação, o programa cliente solicita ao servidor para enviar uma mensagem à estação de gerência. O trecho de programa abaixo ilustra como o programa cliente acessa o programa servidor.

```
/*************************************************************************/
/*  Cria o socket para receber solicitacao dos clientes - 1488           */
/*************************************************************************/

      sock = socket(AF_INET,SOCK_DGRAM,0);
      if ( sock < 0 ) {
            perror("Error - socket datagram creation");
            exit(1);
      }

/*************************************************************************/
/*    Associa um nome ao socket que recebe solicitacao dos clientes      */
/*************************************************************************/

      name_serv.sin_family = AF_INET;
      name_serv.sin_addr.s_addr = INADDR_ANY;
      name_serv.sin_port =1488;
       if (bind(sock,(struct sockaddr_in *)&name_serv,sizeof(name_serv)) < 0){
            perror(" Error - biding a name to socket - ");
            exit(1);
      };
```

Atualmente, está sendo estudada uma maneira de acessar o servidor SNMP do SNM (o snmpd), para que a notificação seja enviada diretamente à Plataforma de Gerência do protótipo.

Capítulo 7

Conclusões

7. Conclusões

A necessidade de uma gerência de redes mais eficiente vem sendo suprida por novas ferramentas e técnicas em sistemas distribuídos. Neste contexto, tem sido discutida uma nova abordagem de gerência de redes, a gerência pró-ativa. Na gerência pró-ativa os problemas da rede são detectados (com base em uma *baseline*) e evitados (com o uso de ações gerência), antes que a rede tenha o seu funcionamento degradado.

O presente trabalho propõe uma arquitetura (ou modelo, discutido na Seção 5.1) para o desenvolvimento da gerência pró-ativa de redes [DEF96][DEF96a] com o auxílio de técnicas de simulação. Este modelo é composto por duas entidades principais, uma local (a Estação de Gerência) e outra remota (o Módulo Pró-ativo, composto por um Serviço de Verficação, uma *Baseline* e um Monitor Remoto), e ambas trocam informações através de um Serviço de Comunicação (desenvolvido com o auxílio de *sockets*). A partir desta arquitetura é possível desenvolver aplicações pró-ativas [DEF95a][DEF95b] e adaptá-las aos sistemas de gerência disponíveis no mercado (de comportamento reativo, como por exemplo, o *SunNet Manager* da *Sun Connect*, o *OpenView* da HP e o NetView AIX/6000 da IBM). Este modelo permite o desenvolvimento de aplicações nas cinco sub-áreas de gerenciamento definidas pela OSI/ISO: performance, falhas, configuração, contabilização e segurança.

Através de uma aplicação de gerência de desempenho e da arquitetura proposta foi implementado um protótipo experimental. Foram realizadas, com o auxílio de um agente remoto e da RMON MIB [WAL91][WAL95], monitorações de parâmetros de performance para compor o perfil da rede. Com base neste perfil foi possível identificar algumas tendências de degradação de desempenho (em parâmetros como, a taxa de utilização, a vazão e a eficiência da rede) no ambiente de testes. Apesar da rede apresentar um bom desempenho durante os testes, foi possível observar algumas características do comportamento pró-ativo com a execução do protótipo (ver Anexo IV). Uma das características observada foi o recebimento das mensagens na Estação de Gerência indicando qual parâmetro estava tendendo a uma

degradação no desempenho. Desta forma foi possível realizar uma ação e impedir a queda do estado ativo do segmento gerenciado.

O desenvolvimento deste trabalho faz parte de um tema atual que ainda é muito discutido por pesquisadores em gerência de redes. O principal objetivo é fornecer uma solução para um problema de interesse real: uma administração de redes mais eficiente. Além disso, este trabalho faz parte do **Projeto PLAGERE ProTeM-CC-II**, Plataformas para Gerência de Redes. Entre os objetivos destacam-se: o desenvolvimento de uma plataforma genérica e de aplicações de gerência de redes. Este projeto possui caráter multi-institucional e tem o apoio do CNPq. As instituições que participam do PLAGERE são a UFSC, o CEFET-PR, a UFPB e o CPqD da Telebrás.

7.1 Perspectivas futuras

Como perspectivas futuras tem-se a possibilidade de continuidade deste trabalho, através das seguintes atividades:

- avaliar o desempenho da rede após a implantação do protótipo, verificando o tipo de sobrecarga gerado pelo modelo;
- desenvolver uma metodologia para o desenvolvimento de *baselines* com o auxílio de ferramentas de simulação;
- implementar um sistema de correções automáticas com o auxílio de sistemas especialistas (*troubletickets*), ou implantar um sistema já desenvolvido [MAD94].
- desenvolver novas aplicações de gerência (nas áreas de falhas, configuração, contabilização, segurança, e até mesmo novas aplicações de performance);
- estender o trabalho para outros tipos de rede, tais como: *Token Ring*, FDDI e ATM;

O presente trabalho teve como meta principal viabilizar a gerência pró-ativa de redes visando melhorar o desempenho de equipamentos, fornecendo uma melhor qualidade dos serviços oferecidos aos usuários. É uma contribuição ao

desenvolvimento de **ferramentas mais eficientes** aplicadas a gerência de redes, que permitem despertar o interesse comercial, disseminando a abordagem pró-ativa fora do ambiente acadêmico.

Referências Bibliográficas

Referências Bibliográficas

[ALM79] ALMES, G.T.; LAZOWSKA, E.D. "The Behaviour of Ethernet-Like Computer Communication Networks". In: *Proceedings of the 7th Symposium on Operating Systems Principles,* ACM-SIGCOMM, Asilomar, California, Dez., 1979. p.66-81.

[ANG95] ANGELO, F.E.V.; DE FRANCESCHI, A.S.M.; WESTPHALL, C.B. "Monitoração da RMON MIB através da Plataforma de Gerência de Redes *SunNet Manager". Relatório de Pesquisa do Projeto PLAGERE*, PROTEM-CC-II/CNPq, dez., 1995.

[ART94] ARTOLA, E.S. "Avaliação da Degradação dos Serviços para Realizar uma Gerência Pró-ativa em Redes". (Trabalho Individual, CPGCC), TI nº 385, CPGCC - UFRGS, fev., 1994.

[BOG88] BOGGS, D.R.; MOGUL, J.C; KENT, C.A. "Measured Capacity of an Ethernet: Myths and Reality". In: *Procedings of the SIGCOMM'88 Symposium on Communications Architectures and Protocols*, ACM SIGCOMM, Stanford, California, Aug., 1988.

[BRU95] BRUNO, L. "O Armazenamento de Redes fica Inteligente". *Byte Brasil*, jan., 1995. p.12-14.

[CAR94] CARRISSIMI, L. S.; BASTOS, E. L.; WESTPHALL, C. B. "Definição de Gerentes e Agentes Para Gerência de Redes". In: XXI *Seminário Integrado de Software e Hardware* (1994 : Caxambu, MG), Caxambu, MG, 1994. p. 397-410.

[CAS93] CASE, J., McCLOGHRIE; K., ROSE, M.; WALDBUSSER, S. "Protocol Operations for version 2 of the Simple Network Management Protocol (SNMPv2)". (*Request for Comments* 1448), Carnegie Mellon University, Apr., 1993.

[COM88] COMER, D.E. "*Internetworking With TCP/IP Principles, Protocols and Architecture*". Vol. I, New Jersey, USA : Prentice-Hall International Inc., 1988.

[COM91] COMER, D.E.; STEVENS, D.L. "*Internetworking With TCP/IP Volume II Design, Implementation and, Internals"*. Vol. II, New Jersey, USA : Prentice-Hall International Inc., 1991.

[COM93] COMER, D.E., STEVENS, D.L. "*Internetworking With TCP/IP Volume III Client-Server Programming and Applications*". Vol. III, New Jersey, USA : Prentice-Hall International Inc., 1991.

[DEF94] DE FRANCESCHI, A.S.M.; NOTARE, M.S.M.A.; RISO, B.G. "Combinando ferramentas de auxílio a utilização de LOTOS para o desenvolvimento de protocolos". Aceito e não publicado In: *INFONOR'94*, Universidad Catolica del Norte, Antofagasta, Chile, 1994.

[DEF95] DE FRANCESCHI, A.S.M.; WESTPHALL, C.B. "Desenvolvimento de Novas Funções de Gerência Utilizando a Plataforma de Gerência *SunNet Manager". Relatório de Pesquisa do Projeto PLAGERE*, PROTEM-CC-II/CNPq, mai., 1995.

[DEF95a] DE FRANCESCHI, A.S.M.; WESTPHALL, C.B. "Avaliação de Desempenho para a Gerência Pró-ativa de Redes". *Relatório de Pesquisa Projeto PLAGERE*, PROTEM-CC-II/CNPq, mai., 1995.

[DEF95b] DE FRANCESCHI, A.S.M.; NOTARE, M.S.M.A.; RISO, B.G. "Uso de ferramentas LOTOS no desenvolvimento de um agente remoto". In: *II CIT - II Congresso de Informática e Telecomunicações*, Cuiabá, MT, set., 1995.

[DEF96] DE FRANCESCHI, A.S.M.; WESTPHALL, C.B. "Avaliação de Desempenho para a Gerência Pró-ativa de Redes". In: *TELEXPO'96, 6º Congresso Internacional de Telecomunições e Teleinformática*, São Paulo, SP, 23-26 mar., 1996.

[DEF96a] DE FRANCESCHI, A.S.M., KORMANN, L.F., WESTPHALL, C.B. "Performance Evaluation for Proactive Management Network". In: *Proceedings of the IEEE/ICC'96, International Conference on Communications*, Dallas, Texas, USA, Jun. 23-27, 1996.

[DEF96b] DE FRANCESCHI, A.S.M.; KORMANN, L.F.; WESTPHALL, C.B. "Performance Application for Proactive Management Network". In: *Proceedings of the IEEE Second International Workshop on Network Management Systems*, Toronto, Canadá, Jun. 19-21, 1996.

[DNP94] DNPAP Group. "Remote Network Monitoring MIB, RFC1271". ***(ftp dnpap.et.tudelft.nl /pub/Whitepaper.ps***) Delft University of Technology, The NetherLands, 1994.

[ERL93] ERLINGER, M. A. "RMON From Concept to Specification". In: *Proceedings of the IFIP/IEEE Integrated Network Management*. H.-G. Hegering and Y. Yemini (editors), Elsevier Science Publishers B.V., North Holland, 1993.

[FER78] FERRARI, D. "*Computer Systems Performance Evalution*". New Jersey, EUA : Prentice-Hall, Inc., 1978.

[GOL93] GOLDSZMIDT, G; YEMINI, Y. "Evaluation Management Decisions via Delegation". In: *Proceedings of the IEEE/IFIP International Symposium on Network Management*, Apr., 1993.

[MOU94] MOUTINHO, C.M.P.; STANTON, M.A. "Aplicações de Gerenciamento de Redes Inteligentes". In: *Anais do 12º SBRC, XXII Simpósio Brasileiro de Redes de Computadores*, Curitiba, 1994.

[NEU93] NEUMAIR, B. "Modelling Resources for Integrated Performance Management". In: *Proceedings of the IFIP/IEEE Integrated Network Management.* H.-G. Hegering and Y. Yemini (editors), Elsevier Science Publishers B.V., North Holland, 1993.

[NOT95] NOTARE, M.S.M.A. "Uma metodologia para a especificação formal de serviços e protocolos de comunicação". (Dissertação de Mestrado), CPGCC, UFSC, abr., 1995.

[QUI95] QUILANTE, M., KURTEN, R., WESTPHALL, C.B. "Novas Funções de Gerência de Segurança usando o AIX NetView/6000". *Relatório de Pesquisa do Projeto PLAGERE*, PROTEM-CC-II/CNPq, set., 1995.

[PRA95] PRAS, A. "Network Management Architectures". (PH. D-thesis nº 95-02), Centre for Telematics and Information Technology, University of Twente, Enschede, The Netherlands, Fev., 1995.

[PER95] PERROW, G.S.; HONG, J.W.; LUTFIYYA, H.L.; BAUER, M.A. "The Abstraction and Modelling of Management Agents". In: *Proceedings of the IEEE/ISINM'95, Fourth IFIP/IEEE International Symposium on Integrated Network Management*, Santa Barbara, CA, USA, 1995. p.466-477.

[REI94] REINHARDT, A. "The Network With Smarts". *Byte*, p.51-64, Oct., 1994.

[ROC94] ROCHA, M.A. "Uma Estratégia para Implementar Gerência Pró-ativa de Redes usando Inteligência Artificial". (Trabalho Individual, CPGCC), TI nº 395 CPGCC - UFRGS, fev., 1994.

[ROC94a] ROCHA, M.A., WESTPHALL, C.B. "Gerência de redes de computadores através de novos agentes". In: *Anais do 12º SBRC, XII Simpósio Brasileiro de Redes de Computadores*, Curitiba, PR, mai., 1994. p.113-133

[ROS91] ROSE, M.T. "*The Simple Book. An Introduction to Management of TCP/IP based Internets*". New Jersey, USA : PTR Prentice-Hall, Inc., 1991.

[ROS94] ROSE, M.T. "*The Simple Book. An Introduction to Management of TCP/IP based Internets*". 2nd ed., New Jersey, USA : PTR Prentice-Hall Inc., 1994.

[RUB95] RUBIN, I. "Traffic and Performance Management of High Speed Networks". In: *Tutorial 6 - ISINM'95, Fourth IFIP/IEEE International Symposium on Integrated Network Management*, May, 1995.

[GON86] GONSALVES, T.A.; TOBAGI, F.A. "On The Performance Effects of Station Locations And Access Protocol Parameters In Ethernet Networks". *IEEE Transactions Communications* 36 (4); Apr., 1988. Originalmente publicado como Relatório Técnico 86-292, Universidade de Stanford, Jan., 1986.

[HAM88] HAMMOND, J. L., O'REILLY, P. J. P. "*Performance Analysis of Local Computer Networks*". EUA : Addison-Wesley, 1988.

[HAY93] HAYES, S. "Analysing Network Performance Management". *IEEE Communications Magazine*, May, 1993.

[HUI90] HUI, J. Y. "*Switching and Traffic Theory for Integrated Broadband Networks*". USA : Kluwer A. Publishers, 1990.

[HUS94] HUSSELBAUGH, B. "(MIS) Using Bandwidth". *Byte*, Dez., 1994. p.117-124

[ISO88] "ISO: *Information processing systems - Open Systems Interconnection - LOTOS - A formal description technique based on the temporal ordering of observational behaviour*", 1988, p. 147.

[JAI91] JAIN, Raj. "*The Art of Computer Systems Performance Analysis*". EUA : John Wiley & Sons, 1991.

[JAN93] JANDER, M. "Proactive LAN Management". *Data Communications*, Mar., 1993. p.49-55.

[KAY94] KAY, R. "Network Management". *Byte*, Dez., 1994. p.104-105.

[KOR94] KORZEIOWSKI, Paul. "Monitoring Your Net". *Byte*, Dez., 1994. p.109-114

[MCK88] MCKERROW, P. "*Performance Measurement of Computer Systems*". Grã-Bretanha : Addison-Wesley, Inc., 1988.

[MAD94] MADRUGA, E.L. "Ferramentas de Apoio à Gerência de Falhas e Desempenho em Contexto Distribuído". (Dissertação de mestrado), CPGCC, UFRGS, Porto Alegre, 1994.

[MAN93] MANSOURI-SAMANI, M.; SLOMAN, M. "Monitoring Distributed Systems". *IEEE Network*, vol. 7, n°6, Nov., 1993. p.20-30.

[MAN95] MANIONE, R.; MONTANARI, F. "Validation and Extension of Fault Management Applications through Environment Simulation". In: *Proceedings of the IEEE/ISINM'95 Fourth IFIP/IEEE International Symposium on Integrated Network Management*, Santa Barbara, CA, USA, 1995. p.466-477.

[MET76] METCALFE, R.M.; BOGGS, D.R. "Ethernet: Distributed Packet Switching for Local Computer Networks". *Communications of ACM*, 19(7):395-404, Jul., 1976.

[SIL93] SILVA, A.C.B., WESTHPALL, C.B. "Implementação de Novos Agentes para Gerência de Redes". In: *Anais do 11º SBRC, XXI Simpósio Brasileiro de Redes de Computadores*, 1993.

[SPO93] SPOHN, Marcelo. "Um Paradigma Orientado a Análise de Performance de Redes de Computadores". (Dissertação de Mestrado), CPGCC, UFRGS, Porto Alegre, mar., 1993.

[STA94] STALLINGS, W. "Packet Filtering in the SNMP Remote Monitor". *Dr. Dobb's Journal*, Nov., 1994.

[STO93] STOCKMAN, B. "A Model for Common Operacional Statistics". (*Request for Comments1404*), Jan., 1994.

[SUN89] SunMicrosystems Inc. "SunNet Manager - Installation and User's Guide". USA, 1989.

[SUN89a] SunMicrosystems, Inc. "SunNet Manager - How Write an Agent". USA, 1989.

[SUN90] SunMicrosystems, Inc. "Network Programming", USA, Mar., 1990. p. 219-316.

[SUN93] Sun Connect, Sun Microsystems, Inc. SunNet Manager 2.2. Programmer's Guide. USA, 1993.

[SUN93a] SunConnect, Sun Microsystems, Inc. SunNet Manager 2.2 Reference Manual. USA, 1993.

[TAN88] TANENBAUM, A.S. "*Computer Networks*". 2nd ed., The Netherlands : Prentice-Hall International, Inc., 1988.

[TAN94] TANENBAUM, A.S. "Redes de Computadores". 2ª edição americana, tradução de PubliCare Serviços de Informática. Rio de Janeiro : Campus, 1994.

[WAL91] WALDBUSSER, S. "Remote Network Monitoring Management Information Base". (*Request for Comments* 1271), Nov., 1991.

[WAL95] WALDBUSSER, S. "Remote Network Monitoring Management Information Base". (*Request for Comments* 1757), Jan., 1995.

[WES88] WESTPHALL, C.B. "Proposição de funções em equipamentos para gerência de comunicação de dados". (Dissertação de Mestrado), CPGCC, UFGRS, Porto Alegre, 1988.

[WES91] WESTPHALL, C.B. "Conception et dévelopment de l'architecture d'adminstration d'un réseau métropolitan". (Thèse de Doctorat nouveau régime), Université Paul Sabatier, juillet., 1991.

[WES92] WESTPHALL, C.B.; ASSOUL, S. "Management Architecture for Networks of the Future". In: *Proceedings of the IFIP/IEEE International Workshop on Distributed Systems : Operations & Management*, Munich, Germany, Oct. 12-13, 1992.

[WES93] WESTPHALL, C.B.; BENSO DA SILVA, A.C. "Desenvolvimento e Integração de Agentes para a Gerência de Redes Locais". Porto Alegre : *Relatório de Pesquisa nº 209*, CPGCC da UFRGS, jan., 1993.

Índice Remissivo

Índice Remissivo

—T—

—U—

Glossário

Glossário

(em ordem alfabética)

baseline

(perfil) É o perfil da rede, que descreve uma caracterização estatisticamente válida do comportamento normal da rede sobre um período estendido de tempo.

bps

(*bits per second* - bits por segundo) Uma medida da taxa de transmissão de dados.

bridge

(Ponte) Um roteador que conecta duas ou mais redes e transmite pacotes entre elas. Normalmente, operam a nível físico, e sua função engloba o armazenamento e envio de pacotes completos.

broadcast

(Difusão) Sistema de entrega de pacotes que entrega uma cópia de um pacote a todos os equipamentos ligados ao domínio de difusão. Pode ser implementado com *hardware* (por exemplo, *Ethernet*) ou por *software* (por exemplo, como um *Cypress*).

CCITT

(Consultative Commitee on International Telephony and Telegraphy) Uma organização internacional que formula padrões para a interconexão de equipamentos telefônicos. Atualmente chama-se ITU.

checksum

É um valor inteiro computado a partir de uma seqüência de octetos, é utilizado para a detecção de erros.

CRC

(Cyclic Redundancy Code) É um valor inteiro computado a partir de uma seqüência de octetos É utilizado para a detecção de erros na transmissão de seqüências de octetos de uma máquina a outra. Normalmente, gasta mais processamento, porém detecta mais erros do que um *checksum*.

CSMA/CD

(Carrier Sense Multiple Access with Collision Detection) Uma característica do *hardware* da rede para controlar o acesso ao meio, o qual detecta colisões quando duas estações tentam transmitir simultaneamente.

Ethernet

É uma das tecnologias de redes mais populares que existem. Foi inventada pela *Xerox Corporation Palo Alto Research Center*. Utiliza a tecnologia CSMA/CD.

FDDI

(Fiber Distribution Data Interface) Padrão para a tecnologia de redes baseada em fibras óticas proposto pelo *American National Standards Institute* (ANSI).

frame

(quadro) É literalmente, um pacote quando transmitido através de uma linha serial.

FTP

(File Transfer Protocol) Protocolo de aplicação para a transferência de arquivos da *Internet*.

gateway

Computador dedicado que liga duas ou mais redes e faz o roteamento dos pacotes de uma rede à outra.

host

Estação de trabalho ligada a rede, ou outro equipamento ativo que possua uma interface de rede.

hub

Termo normalmente atribuído a um repetidor de múltiplas portas ou concentrador de rede.

ICMP

(Internet Control Message Protocol) Parte integral do protocolo IP que manipula mensagens de controle e erros.

IEC

(International Electrotechnical Commitee).

IETF

(Internet Engineering Task Force) Organização internacional de padronizações. Um dos seus grupos de trabalho desenvolveu a RMON MIB.

IP

(Internet Protocol) Protocolo padrão *Internet* que define o datagrama como unidade de informação passada através da rede, e fornece a base para a entrega de pacotes sem conexão.

ISO

(International Organization for Standartization) Entidade internacional que define, discute, propõe e especifica padrões para protocolos de redes. A ISO é mais conhecida pelo seu modelo de refência que define as sete camadas para a organização conceitual de protocolos.

ITU

(International Telecommunications Union) Antigo CCITT.

LAN

(Local Area Networks) Qualquer tecnologia a nível físico de rede que opere a altas velocidades em distâncias não muito longas.

multicast

Técnica que permite repassar uma cópia de um único pacote a um subconjunto selecionado entre todos os destinos da rede.

OSI

(Open Systems Interconnection) Uma referência de protocolos, especificamente padrões ISO, para a interconexão de sistemas abertos.

packet capture

Captura de pacotes. Um dos nove grupos de objetos da RMON MIB.

PING

(Packet InterNet Groper) Programa utilizado para testar a alcançabilidade de destinatários da rede. Também é o nome de um agente disponível no *SunNet Manager (SNM).*

proxy

(procurador) Nome determinado a agentes que fazem a tradução de um protocolo para outro.Por exemplo, do protocolo do SNM para o SNMP.

RMON MIB

(Remote Monitoring Network Management Information Base) Base de informações para a monitoração remota de redes. Especificada no *Internet Request for Coments (RFC) 1271.*

smart hub

Concentrador de redes com mecanismo inteligente.

SNM

(SunNet Manager) Plataforma de gerência de redes da Sun Connect.

SNMP

(Simple Network Management Protocol) Protocolo da camada de aplicação para o gerenciamento de redes TCP/IP (*Internet*).

socket

Recurso disponível no sistema operacional *SunOS* para permitir que um processo acesse a rede.

TCP

(Transmission Control Protocol) Protocolo a nível de transporte da I*internet,* fornece o serviço confiável com conexão.

TELNET

Protocolo de aplicação Internet que fornece o serviço de conexão de terminal remoto.

throughput

(vazão) Métrica de performance que fornece o número de transações por unidade de tempo.

threshold

(limiar) É um valor utilizado como limiar, normalmente utilizada em sistemas de gerência para o envio de alarmes e notificações de eventos.

transceiver

Dispositivo que conecta uma interface de um equipamento a uma rede local.

troubletickets

Sistema para armazenamento de fichas com problemas de uma rede e suas possíveis soluções.

UDP

(User Datagram Protocol) Protocolo padrão Internet que permite a um programa de aplicação enviar um datagrama a um outro programa em outra máquina.

trap

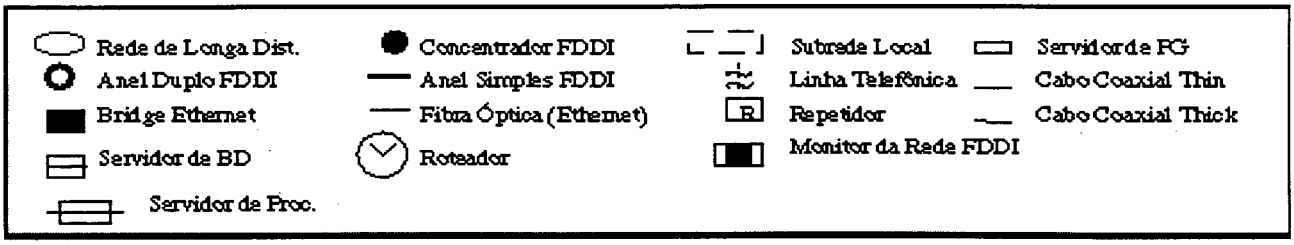
Operação de notificação do protocolo SNMP da *Internet.*

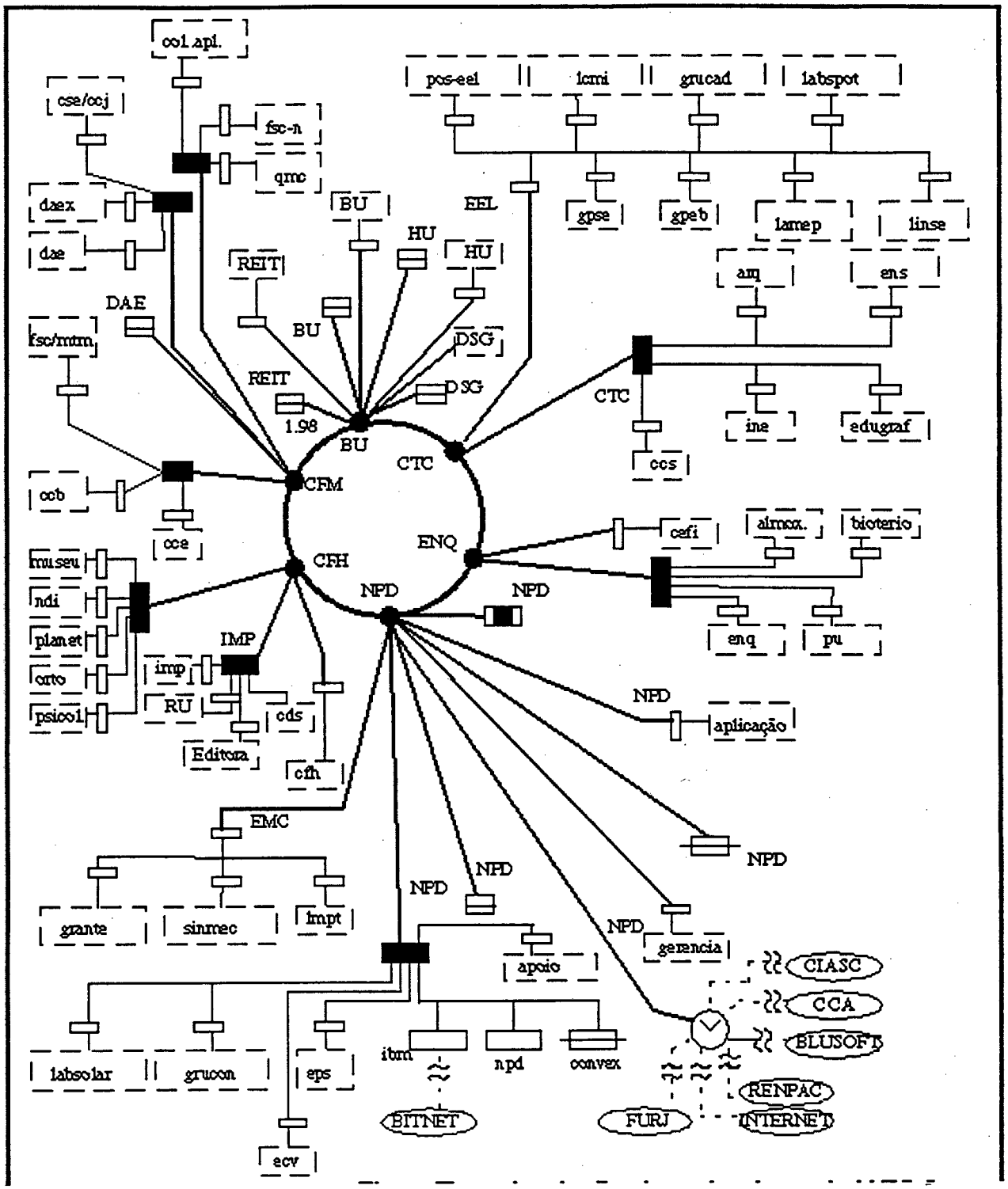
Anexo I

Topologia da Rede UFSC

Este anexo fornece a topologia completa da rede da UFSC. Esta foi a última relação feita nos últimos meses e está disponível no servidor WWW da UFSC, no endereço http:\\www.npd.ufsc.br.

Legenda:




cc1 apl.
cse/ccj
fsc-n
qmc
daex
dae
BU
HU
HU
EEL
REIT
DAE
BU
DSG
fsc/mtm
REIT
DSG
1.98
BU
CTC
CFM
ccb
cce
CFH
ENQ
NPD
NPD
IMP
imp
RU
cds
Editora
cfh
EMC
grante
sinmec
lmpt
NPD
NPD
NPD
NPD
apoio
labsolar
grucon
eps
ibm
npd
convex
ecv
BITNET
FURJ
CIASC
CCA
BLUSOFT
RENPAC
INTERNET
pos-eel
lcmi
grucad
labspot
gpse
gpeb
lamep
linse
am
ens
CTC
ine
edugraf
ccs
cefi
almox.
bioterio
enq
pu
aplicação
NPD
NPD
museu
ndi
planet
orto
psicol

Anexo II
Arquivos Fontes

Arquivo fonte do Serviço de Verificação, que acessa o Monitor Remoto Beholder, filtra as informações do protótipo e utiliza o Serviço de Comunicação. O filtro de informações das estações foi desenvolvido por Henrique Lopes Weber durante a disciplina de Gerência de Redes II ministrada pelo Prof. Carlos Becker Westphall no Curso de Pós-Graduação em Ciência da Computação da UFSC.

```
/* Serviço de Verificacao desenvolvido para o prototipo experimental de*/
/* gerencia pro-ativa by Analucia Schiaffino Morales De Franceschi     */
/* (Filtro de informacoes dos hosts by Henrique Lopes Weber)           */

#include <string.h>
#include <stdio.h>
#include <stdlib.h>
#include <time.h>



#define IAA -1      /* Impossivel Abrir Arquivo */
#define VNEA -2     /* Variavel Nao Existe no Arquivo */

      /* Constantes com as iniciais dos nomes das variaveis */
#define HTIP 0
#define HTOP 1
#define HTIO 2
#define HTOO 3
#define HTOE 4
#define HTOBP 5
#define HTOMP 6
#define HTA 7
#define SUT 8

#define VENUS 0
#define APOLO 1
#define CERES 2
#define JANUS 3
#define ATLAS 4
#define MERCURIO 5
#define GRAD_GW 6

      /* Nome das variaveis */
char NOME_VARS [9][50]= { "hostTimeInPkts",
                          "hostTimeOutPkts",
                        "hostTimeInOctets",
                        "hostTimeOutOctets",
                        "hostTimeOutErrors",
                        "hostTimeOutBroadcastPkts",
                        "hostTimeOutMulticastPkts",
                       "hostTimeAddress",
                           "sysUpTime"};

char END_HOSTS [7][20]={"08:00:20:11:fd:25\0",
                     "08:00:20:0d:2e:8c\0",
                         "08:00:20:04:0f:f9\0",
                         "00:00:3b:80:38:46\0",
                         "08:00:20:0e:9b:66\0",
                         "00:00:3b:80:33:9e\0",
                         "00:20:af:a4:d7:ce\0"};

/*long somatorio(char nome_arq[], int nome_var);

int difftime(time_t time2, time_t time_1);
*/
```

```
if (!strstr (linha_f, NOME_VARS[7]))
{
        printf("var nao encontrada %s\n", NOME_VARS[7]);
       fclose (f);
       return VNEA;
}
       /* Procura enderecos das estacoes desejadas */
 num_l=1;

 do {
 str_host = strchr(linha_f,'=')+1;


 if(strcmp(str_host,END_HOSTS[VENUS])==0){
   indx[i]=num_l;
   i++;
   }
 if(strcmp(str_host,END_HOSTS[APOLO])==0){
    indx[i]=num_l;

    i++;
   }
 if (strcmp(str_host,END_HOSTS[CERES])==0){
    indx[i]=num_l;

    i++;
   }
 if (strcmp(str_host,END_HOSTS[JANUS])==0){
   indx[i]=num_l;

   i++;
   }
 if (strcmp(str_host,END_HOSTS[ATLAS])==0){
   indx[i]=num_l;

   i++;
  }
 if (strcmp(str_host,END_HOSTS[MERCURIO])==0){
   indx[i]=num_l;

   i++;
   }
 if (strcmp(str_host,END_HOSTS[GRAD_GW])==0){
   indx[i]=num_l;

   i++;
   }
 fscanf(f,"%s",linha_f);
 num_l++;

 } while(ftell(f)<tam_f && strstr(linha_f,NOME_VARS[7]));


 do
   fscanf (f, "%s", linha_f);
while (!strstr (linha_f, NOME_VARS [nome_var]) && ftell (f) < tam_f);

/* Se a variavel desejada nao foi encontrada ... */
if (!strstr (linha_f, NOME_VARS [nome_var]))
{
       fclose (f);
       return VNEA;
}
 j=i;
 num_l=1;
```

```
/* Funcao para obter a diferenca entre dois tempos do sistema*/

int difftime(time_t time2, time_t time1)
{
 struct tm *systime;
 int result,t1,t2;
 int tt1,tt2;

 systime=localtime(&time1);
 t1=systime->tm_sec;
 tt1=systime->tm_min;

 systime=localtime(&time2);
 t2=systime->tm_sec;
 tt2=systime->tm_min;

 if(tt2==tt1)
   result=t2-t1;

 if(tt2<tt1){
   t2=60+t2;
   result=t2-t1;}

 return result;
}
*/

/* Funcao para obter o somatorio dos valores do arquivo de monitoracao de dados*/

long somatorio_var (char nome_arq[], int nome_var)
{
 char linha_f[51],
      linha[51],    /* linha lida do arquivo */
      *str_num;     /* string do numero da linha (apos o = da linha) */
char  *str_host;
char  *str_soma;
char  *str_time;
int tam_f;
int indx[10];
int i=0;
int j=0;
int num_l;
long result = 0;
FILE *f;


       if ( !(f = fopen (nome_arq, "r"))) /* Se arquivo nao existe ... */
             return IAA;

             /* Verifica o tamanho do arq e armazena em tam_f. */
       fseek (f, 0, 2);
       tam_f = ftell (f);
       fseek (f, 0, 0);


       /* Procura pela primeira ocorrencia da var desejada no arquivo. */

        fseek(f, 0, 0);
        do
         fscanf (f, "%s", linha_f);
       while (!strstr (linha_f, NOME_VARS[7]) && ftell (f) < tam_f);

             /* Se a variavel desejada nao foi encontrada ... */
```

```
        i=0;
        do{
             if(num_l==indx[i]){
              str_soma=strchr(linha_f,'=')+1;
              result+=atol(str_soma);
              i++;
              }
        fscanf(f,"%s",linha_f);
        num_l++;


        }while(ftell(f)<tam_f && i<=j);

       fclose (f);
       return result;
}


main (int argc, char *argv[])
{
   int    stat, count;
   char   ch;
   time_t x, y;
   int  Rx, Tx, Ry, Ty;
   long unsigned int InRatePkts, OutRatePkts;
   long unsigned int InRateOctets, OutRateOctets;
   long unsigned int LineUtil;


   /* Calcula a taxa de entrada e de saida em pacotes */
  printf("\n\n\n");
  printf("_/ _/ _/ _/ _/ _/ _/ _/ _/ _/ _/ _/ _/ _/ _/\n");
  printf("           Monitoracao do Prototipo          \n");
  printf("_/                                          _/\n");
  printf("             de Gerencia Pro-ativa           \n");
  printf("_/                                          _/\n");
  printf("      Analise de Desempenho Utilizando       \n");
  printf("_/                                          _/\n");
  printf("              um Agente RMON - BTNG          \n");
  printf("_/ _/ _/ _/ _/ _/ _/ _/ _/ _/ _/ _/ _/ _/ _/\n");
  printf("  Estacoes:  venus - apolo - ceres           \n");
  printf("_/    atlas - janus - mercurio - grad-gw  _/ \n");
  printf("                                            \n");
  printf("_/ _/ _/ _/ _/ _/ _/ _/ _/ _/ _/ _/ _/ _/ _/\n");
  printf("\n\n\n");

do{
   x = time(NULL);
   stat = system ("echo 'hostTimeEntry[]...'|snmp-tbl > tt1");
   Rx = somatorio_var ("tt1", HTIP);
   Tx = somatorio_var ("tt1", HTOP);

   y = time(NULL);
   stat = system ("echo 'hostTimeEntry[]...'|snmp-tbl > tt2");
   Ry = somatorio_var ("tt2", HTIP);
   Ty = somatorio_var ("tt2", HTOP);

   InRatePkts = (Ry-Rx)/(difftime(y,x));
   OutRatePkts= (Ty-Tx)/(difftime(y,x));

   printf("Taxa de entrada: %d pkts/s\n",InRatePkts);
   printf("Taxa de saida  : %d pkts/s\n",OutRatePkts);
   printf("Throughput: %d pkts/s\n",InRatePkts+OutRatePkts);
   stat=system("rm tt1");
   stat=system("rm tt2");
```

Sistema de Comunicação utilizando recursos de rede e sistemas distribuídos (*sockets*). Este arquivo foi retirado do trabalho de Projeto II do Curso de Graduação em Ciência da Computação da UFSC, que teve como orientador o professor Carlos Becker Westphall e como co-orientadora Analúcia Schiaffino Morales De Franceschi.

```
/*************************************************************************/
/*  send TRAP - Fabio Eduardo & Analucia De Franceschi                   */
/*************************************************************************/

#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <netdb.h>
#include <stdio.h>


main(argc, argv)
       int argc;
       char *argv[];
{

       int sock;
       struct sockaddr_in name_serv, name_cli;
       struct hostent *hp, *gethostbyname();
       int cli_len, serv_len;
         char msg[1024];
/*************************************************************************/
/*  Cria o socket do cliente que receberá as respostas do servidor       */
/*************************************************************************/

       sock = socket (AF_INET, SOCK_DGRAM,0);
       if ( sock < 0 ) {
              perror ("Error - socket creation - ");
              exit (0);
              }
/*************************************************************************/
/* Associa um nome ao socket do cliente que receberá respostas do        */
/* servidor                                                              */
/*************************************************************************/

       name_cli.sin_family = AF_INET;
       name_cli.sin_addr.s_addr = INADDR_ANY;
       name_cli.sin_port = 0;
       if (bind (sock, (struct sockaddr_in *)&name_cli,sizeof(name_cli))<0)     {
              perror("Error - binding a name to socket");
              exit(0);
       }
       cli_len = sizeof(name_cli);
       if (getsockname(sock, (struct sockaddr_in *)&name_cli, &cli_len)<0){
              perror("obtencao do nome do socket");
                 exit(1);
       }
       printf("porta do socket #%d\n", ntohs(name_cli.sin_port));
/*************************************************************************/
/* GETHOSTBYNAME retorna uma estrutura  onde inclui o endereco da rede   */
/* do host especificado.                                                 */
/*************************************************************************/

       hp = gethostbyname(argv[1]);
       if ( hp == 0 ) {
              fprintf ( stderr, " %s: host unrecognized" , argv[1]);
              exit(0);
        }
```

```
  /* Calcula a taxa de entrada e de saida em bytes */

   x = time(NULL);
   stat = system ("echo 'hostTimeEntry[]...'|snmp-tbl > tt1");
   Rx = somatorio_var ("tt1", HTIO);
   Tx = somatorio_var ("tt1", HTOO);

   y = time(NULL);
   stat = system ("echo 'hostTimeEntry[]...'|snmp-tbl > tt2");
   Ry = somatorio_var ("tt2", HTIO);
   Ty = somatorio_var ("tt2", HTOO);

   InRateOctets = (Ry-Rx)/(difftime(y,x));
   OutRateOctets= (Ty-Tx)/(difftime(y,x));
   printf("-------------------------------------\n");
   printf("Taxa de entrada: %d bytes/s\n",InRateOctets);
   printf("Taxa de saida  : %d bytes/s\n",OutRateOctets);
   printf("Throughput: %d bytes/s\n",InRateOctets+OutRateOctets);

   /* Utilizacao da linha */
   printf("-------------------------------------\n");
   LineUtil = ((InRateOctets+OutRateOctets)*8)/10000;
   printf("Utilizacao da Linha: %d %%\n", LineUtil);
   printf("-------------------------------------\n");

   stat=system("rm tt1");
   stat=system("rm tt2");

   printf("Press <enter> to continue ...\n");
  }while((ch = getchar()) == '\n');

   return 0;
}
```

```
            perror(" Error - biding a name to socket - ");
            exit(1);
      };

/*************************************************************************/
/* Constroi o nome do socket correspondente ao canal de respostas para   */
/*  os clientes                                                          */
/*************************************************************************/

      name_cli.sin_family = AF_INET;

/*************************************************************************/

      while (1){

/*************************************************************************/
/*                 Esperando requisicao do cliente                       */
/*************************************************************************/

            cli_len = sizeof( struct sockaddr_in );
            if (recvfrom(sock,(char *)msg,sizeof(msg),0,(struct
      sockaddr*)&name_cli,&cli_len) < 0){
                  perror("Error - recieving client request");
                  exit(1);}

/*************************************************************************/
/*  Se a requisicao é write, imprime na tela e envia mensagem OK         */
/*************************************************************************/

            printf("%s\n",msg);}


/*************************************************************************/

      close(sock);
      exit(0);
}
```

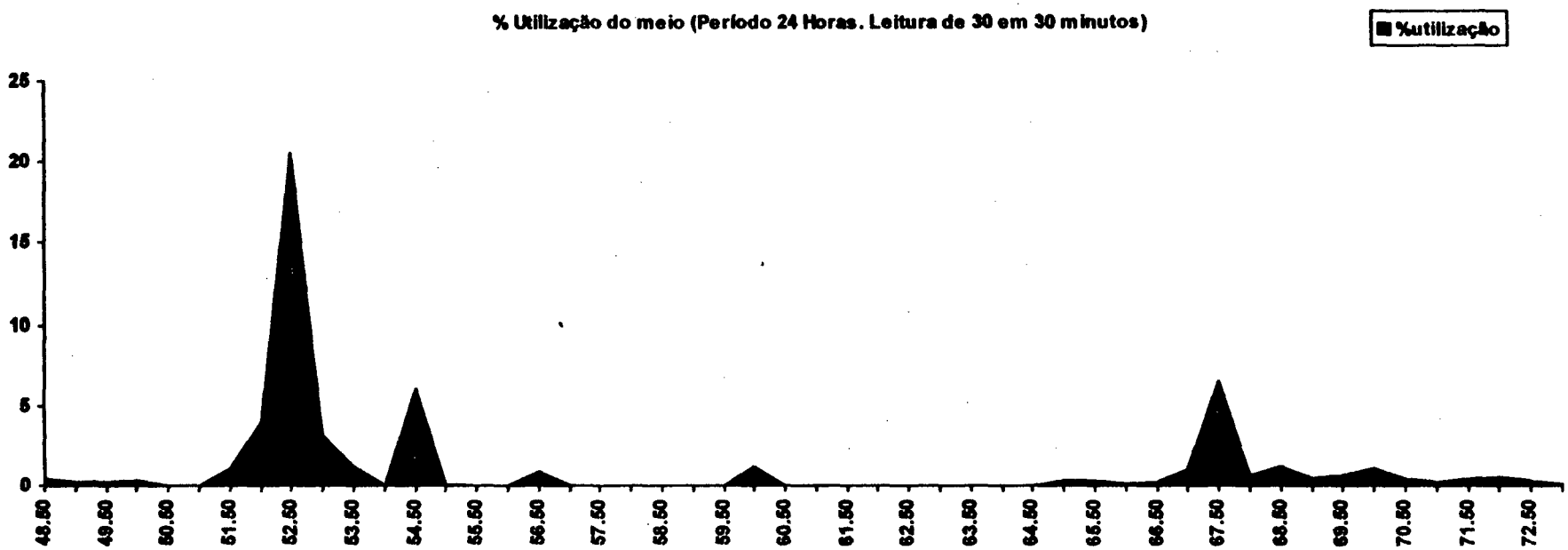
```
/*************************************************************************/
/* Constroi o nome, do socket correspondente ao canal de solicitacao de  */
/*  servicos ao servidor tty.                                            */
/*************************************************************************/

        bcopy(hp->h_addr, &name_serv.sin_addr, hp->h_length);
       name_serv.sin_family = AF_INET;
       name_serv.sin_port = htons(1488);

        strcpy(msg,argv[2]);

       if (sendto(sock,(char *)msg,sizeof msg,0,(struct sockaddr*)&name_serv,sizeof
(name_serv)) < 0)
             {
              perror("Error - sending datagram message");
              exit(0);
            }

       printf("%s\n",msg);



/*************************************************************************/

       close(sock);
       exit(0);
}


/*************************************************************************/
/*             Trap Deamon - Fabio Eduardo  & Analucia De Franceschi     */
/*************************************************************************/

#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <stdio.h>

main()
{
       int sock,lenght;
       struct sockaddr_in name_serv, name_cli;
       int cli_len;
        char msg[1024];


/*************************************************************************/
/*  Cria o socket para receber solicitacao dos clientes - 1488           */
/*************************************************************************/

       sock = socket(AF_INET,SOCK_DGRAM,0);
       if ( sock < 0 ) {
              perror("Error - socket datagram creation");
              exit(1);
       }

/*************************************************************************/
/*    Associa um nome ao socket que recebe solicitacao dos clientes      */
/*************************************************************************/

       name_serv.sin_family = AF_INET;
       name_serv.sin_addr.s_addr = INADDR_ANY;
       name_serv.sin_port =1488;
    if (bind(sock,(struct sockaddr_in *)&name_serv,sizeof(name_serv)) < 0){
```

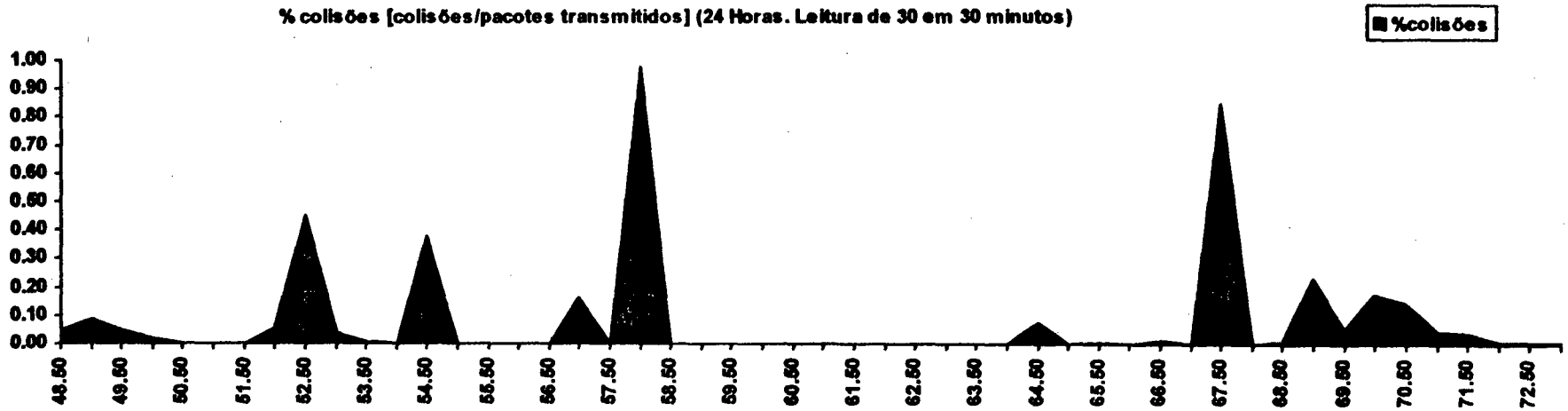
Anexo III
Resultados das Monitorações

A seguir são fornecidas duas amostras de resultados de monitorações interpretados com o auxílio de um banco de dados.

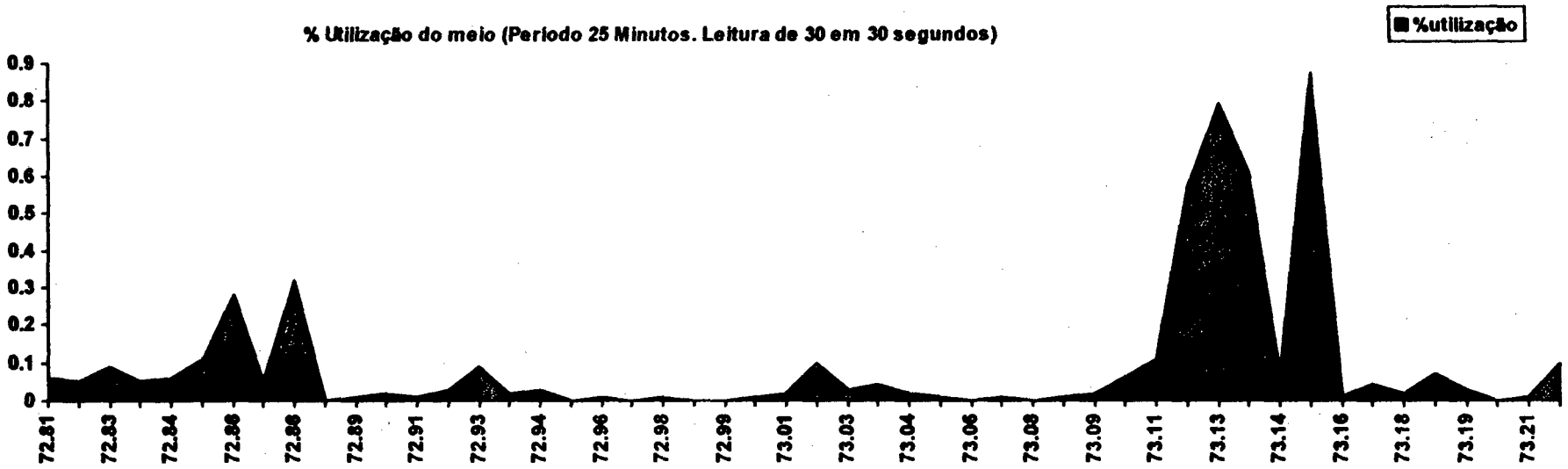
O primeiro grupo de amostras é do Grupo *History* da RMON MIB e o segundo, é do Grupo Hosts. As amostras foram coletadas pelo agente remoto utilizado no trabalho. O formato do armazenamento destas variáveis na MIB é semelhante ao formato das variáveis do Grupo *Filter*, ilustrado como exemplo no Capítulo 5, Seção 5.2.2.3. Através do uso de um Banco de Dados (*Microsoft Access*) foi possível interpretar os resultados das amostras, e calcular algumas métricas de performance descritas no trabalho.

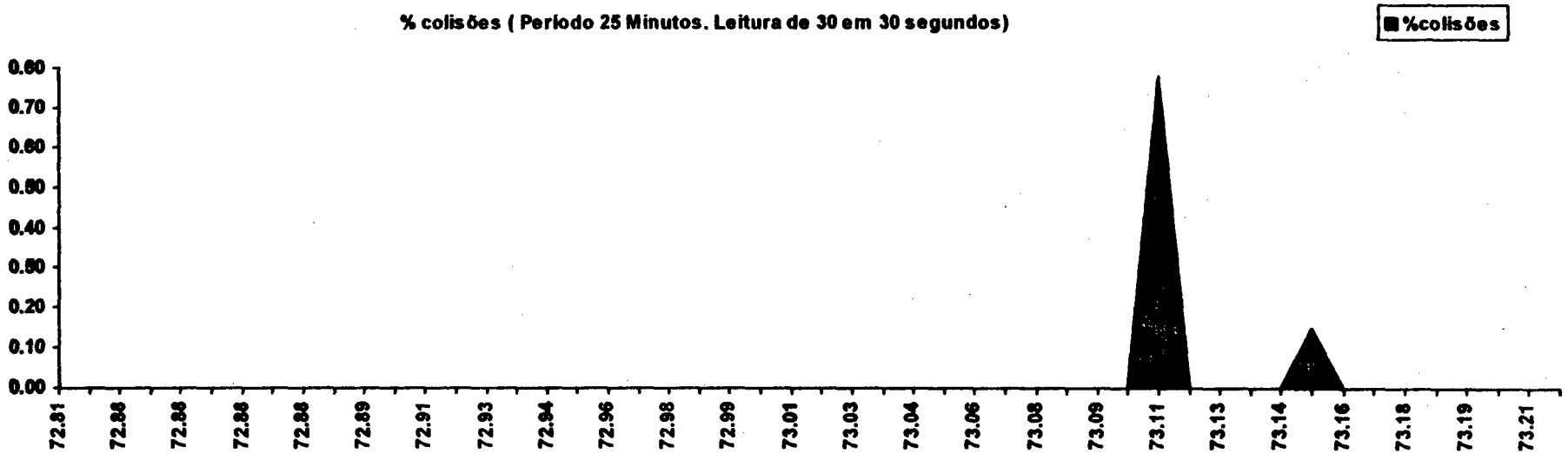
IntervalSta	Octets	Pkts	OverszPkts	UndszPkts	Utilization	BroadcPkts	Collisions	CRCAliErr	DropEvents	%PktsColl	%Utiliz





IntervalSta	Octets	Pkts	OverszPkts	UndszPkts	Utilization	BroadcPkts	Collisions	CRCAliErr	DropEvents	%PktsColl	%Utiliz

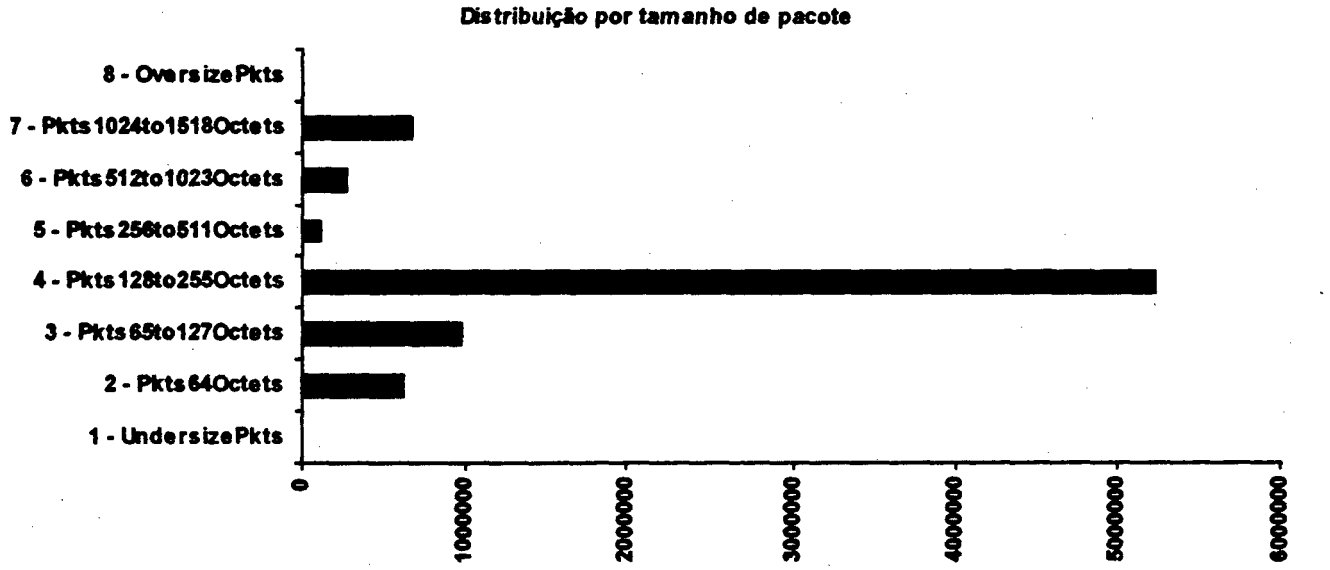




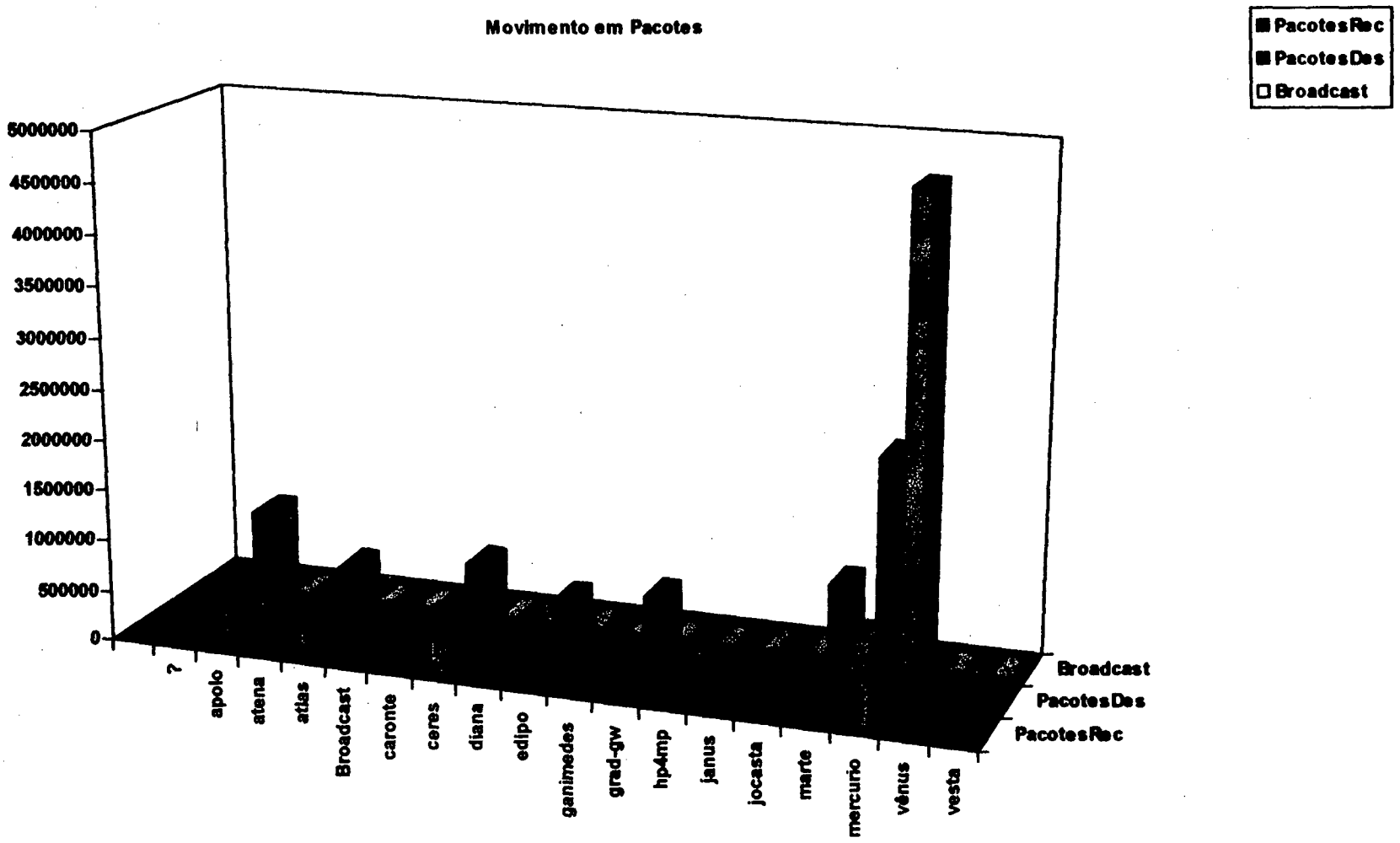
etherStats

Data/hora leitura:	05/01/96 18:00:00	**Tempo de sysUp:**	73:13:59	263579.52

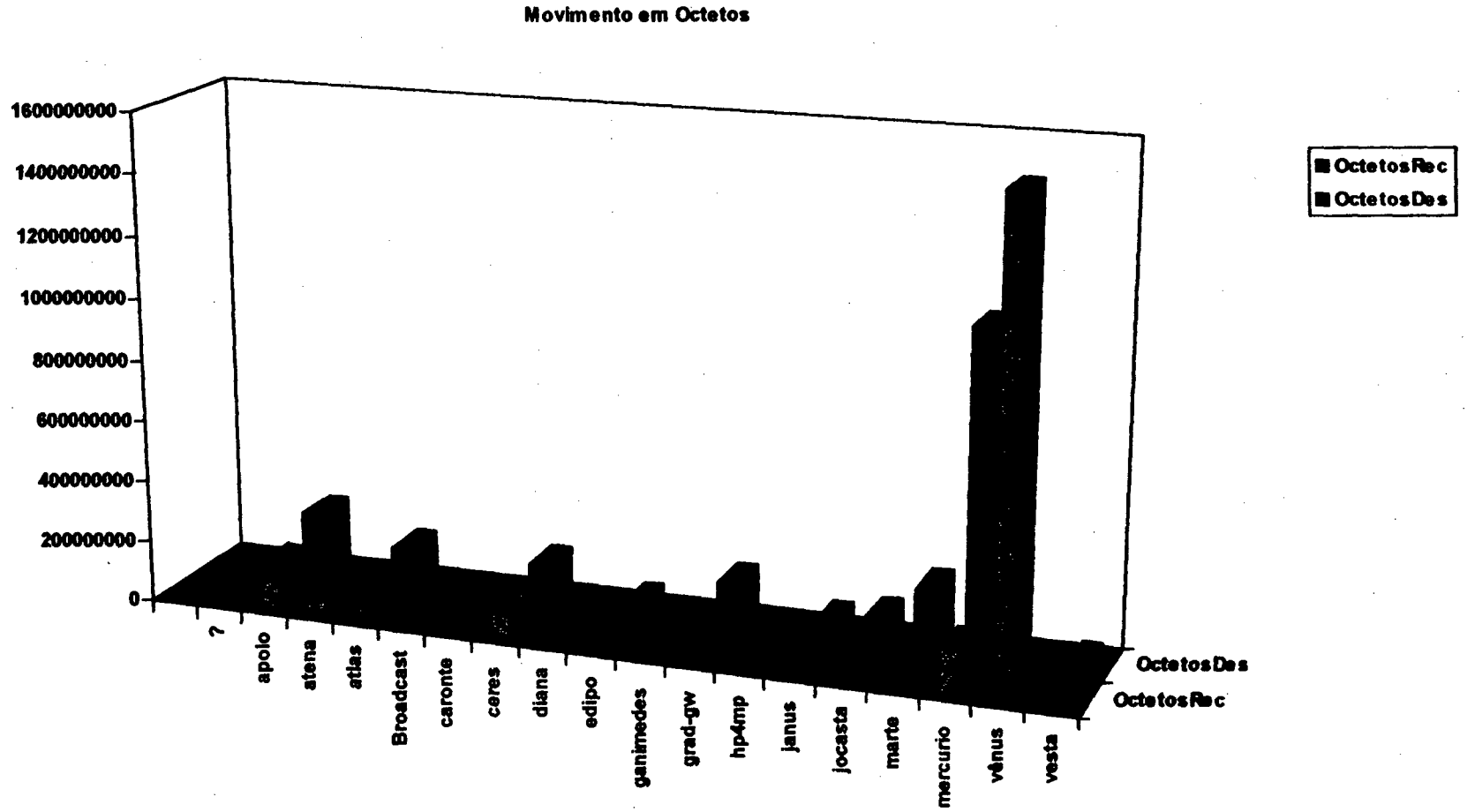
etherStatsUndersizePkts	1069
etherStatsPkts64Octets	614676
etherStatsPkts65to127Octets	967550
etherStatsPkts128to255Octets	5237521
etherStatsPkts256to511Octets	111272
etherStatsPkts512to1023Octets	266277
etherStatsPkts1024to1518Octets	668610
etherStatsOversizePkts	0
etherStatsPkts	7866975
etherStatsBroadcastPkts	30654
etherStatsMulticastPkts	0
etherStatsOctets	2208354093
etherStatsCollisions	23992
etherStatsCRCAlignErrors	0
etherStatsDropEvents	297368
etherStatsFragments	0
etherStatsJabbers	0



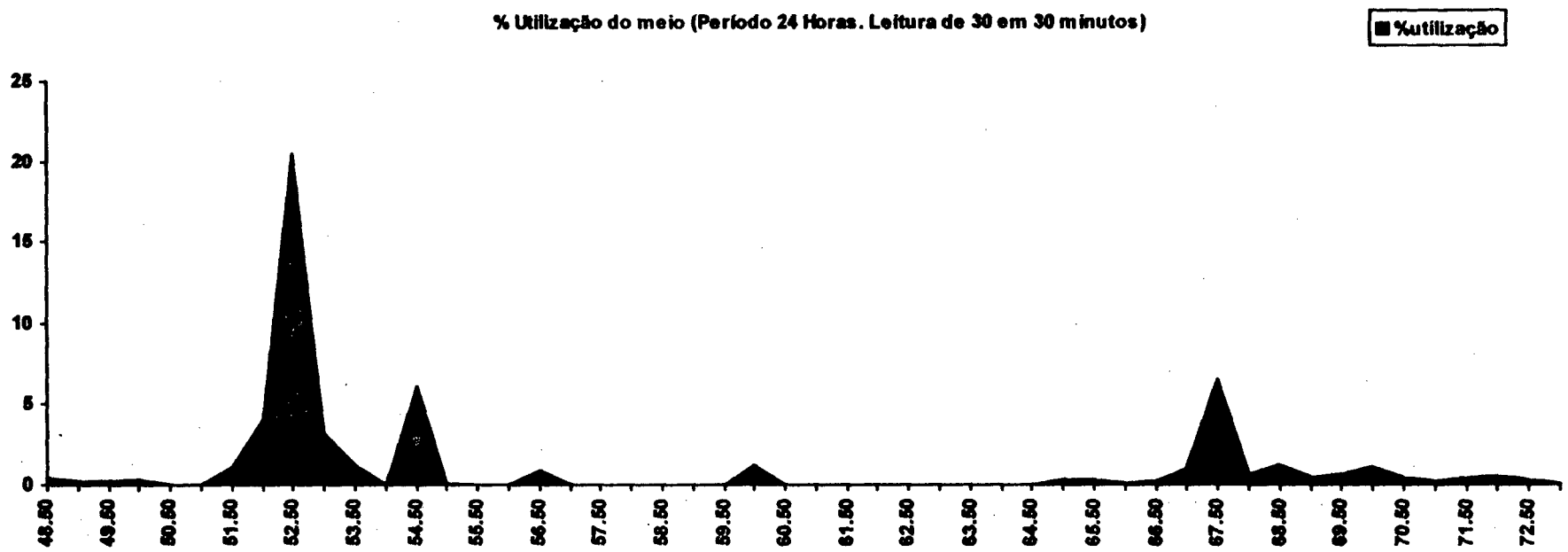
Host Name	Address	InOctets	InOctSeg	OutOctets	OutOctSeg	InPkts	OutPkts	OutBroadcPkts	%OctIn	%OctOut

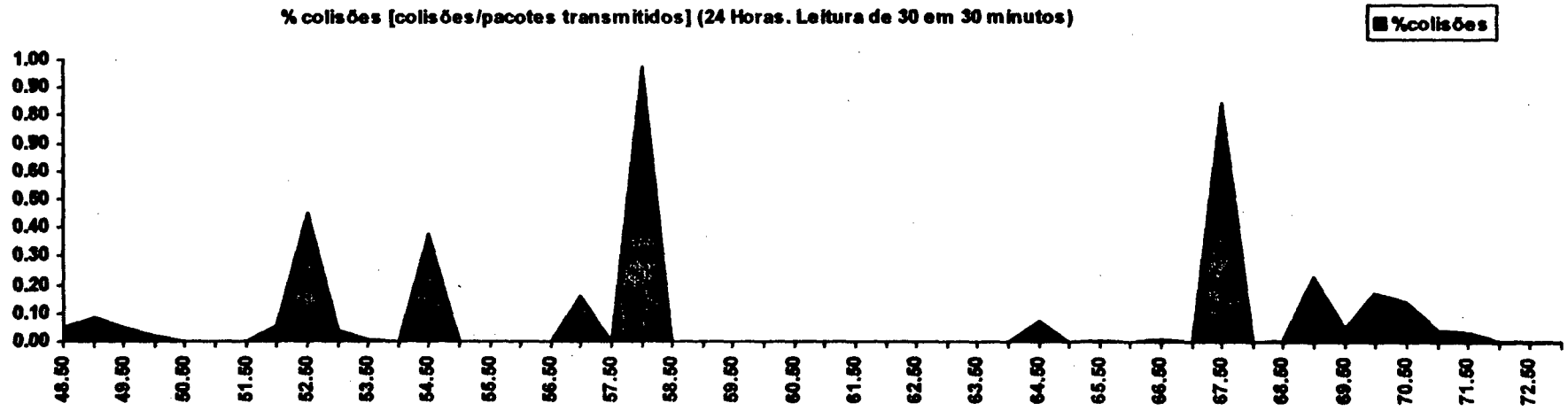


Host Name	Address	InOctets	InOctSeg	OutOctets	OutOctSeg	InPkts	OutPkts	OutBroadcPkts	%OctIn	%OctOut



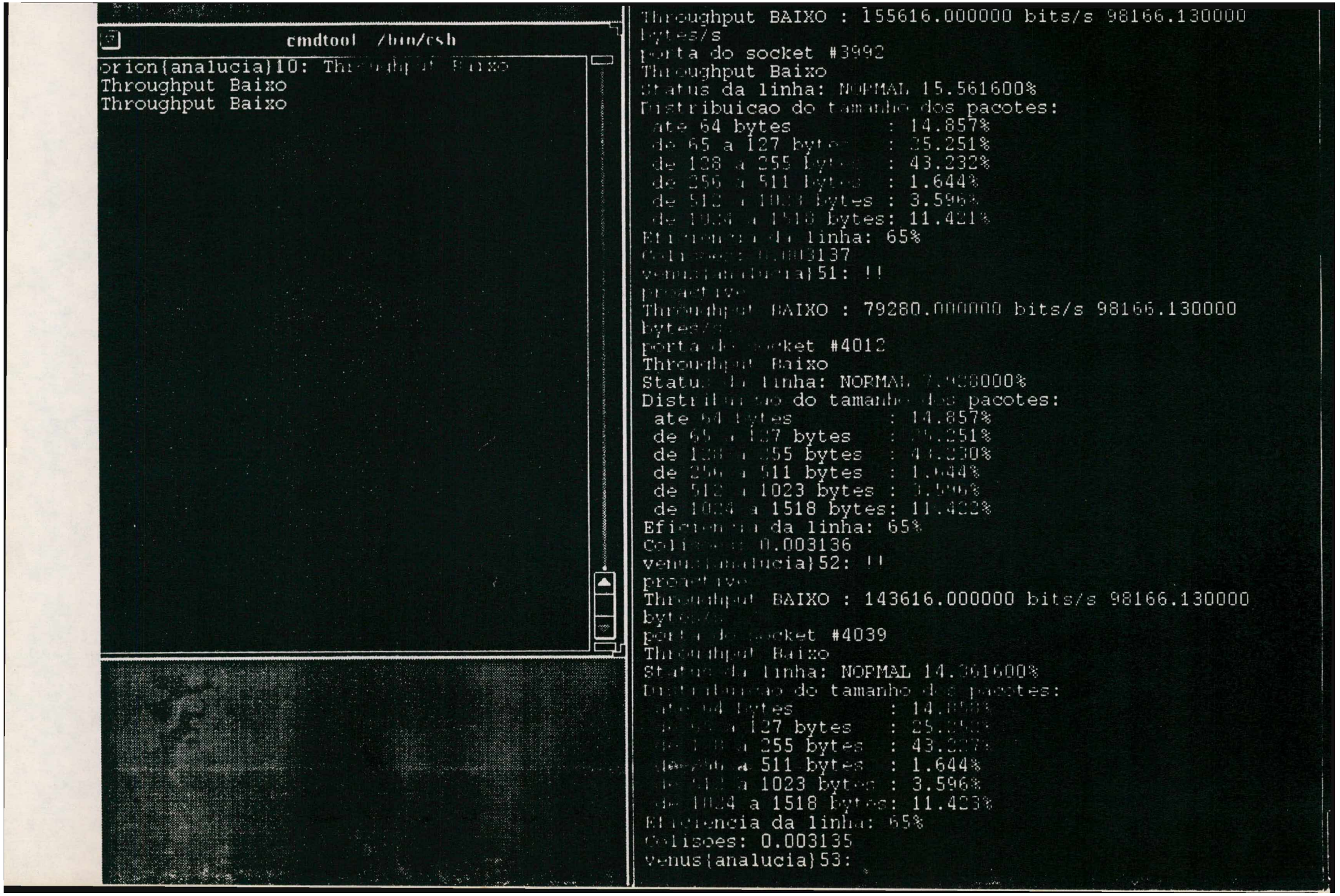
IntervalSta	Octets	Pkts	OverszPkts	UndszPkts	Utilization	BroadePkts	Collisions	CRCAliErr	DropEvents	%PktsColl	%Utiliz

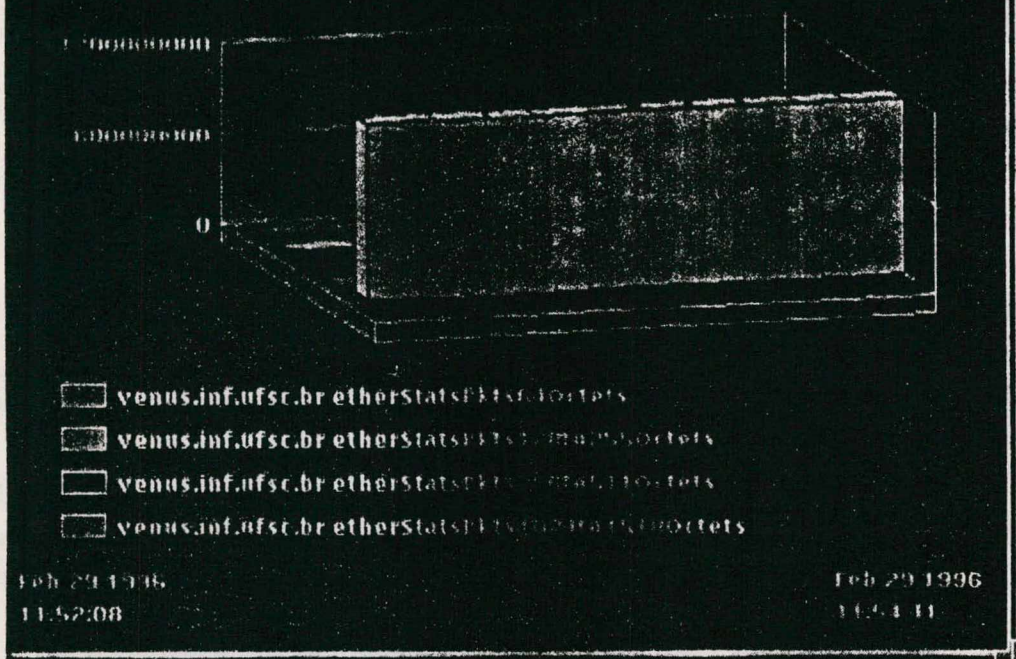


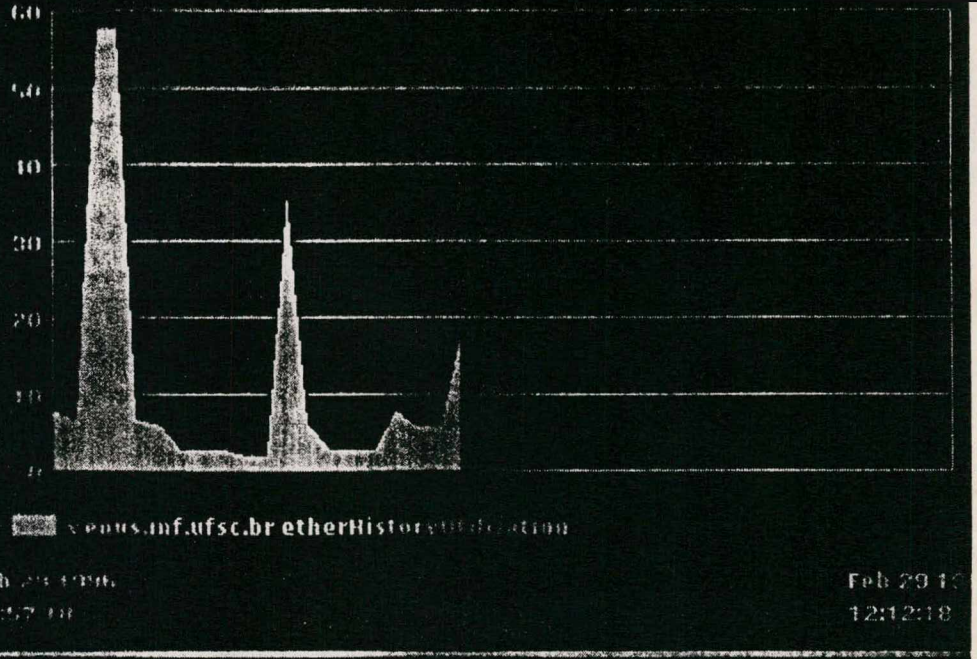


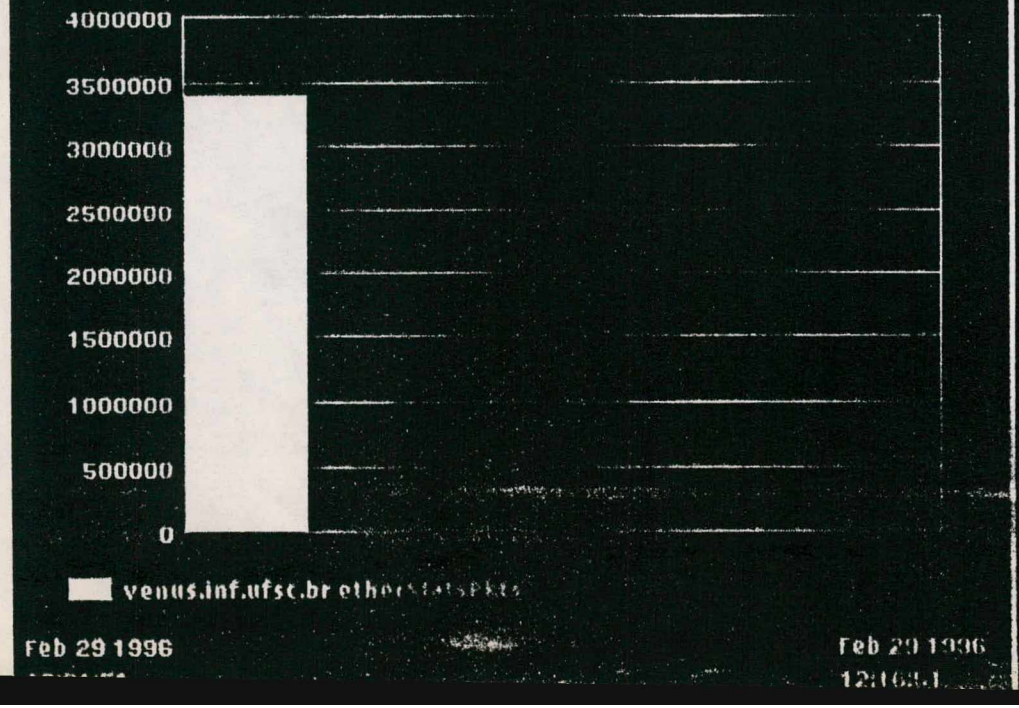
Anexo IV

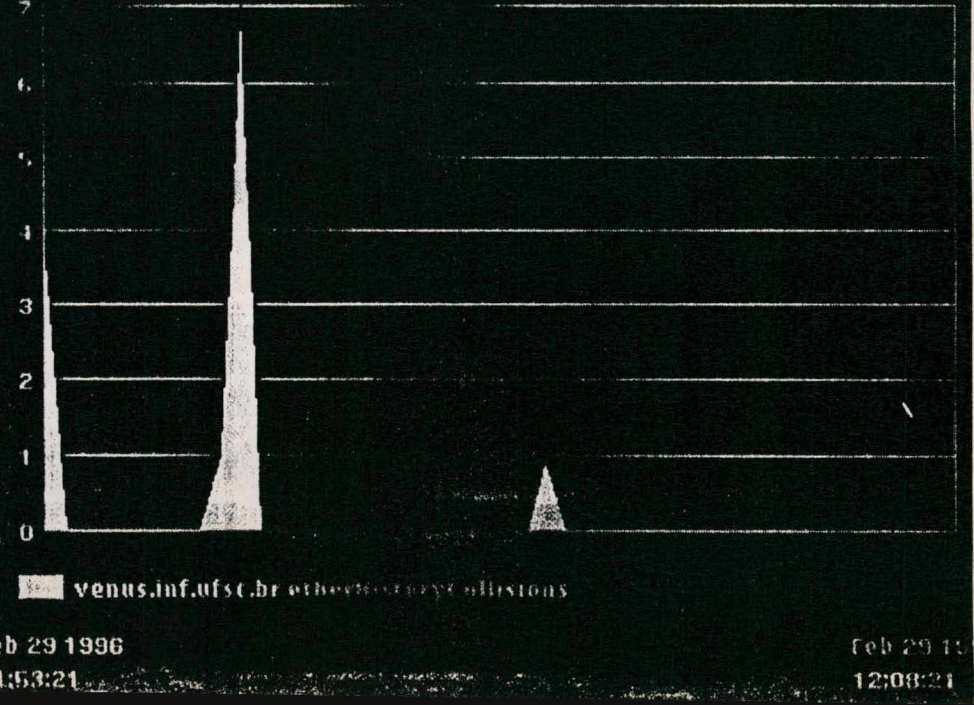
Execução do Protótipo


cmdtool /bin/csh
orion{analucia}10:
Throughput Baixo
Throughput Baixo
Throughput BAIXO : 155616.000000 bits/s 98166.130000
bytes/s
porta do socket #3992
Throughput Baixo
Status da linha: NORMAL 15.561600%
Distribuicao do tamanho dos pacotes:
ate 64 bytes : 14.857%
de 65 a 127 bytes : 25.251%
de 128 a 255 bytes : 43.232%
de 256 a 511 bytes : 1.644%
de 512 a 1023 bytes : 3.596%
de 1024 a 1518 bytes: 11.421%
Eficiencia da linha: 65%
0.003137
venus{analucia}51: !!
Throughput BAIXO : 79280.000000 bits/s 98166.130000
porta do socket #4012
Throughput Baixo
Distribuicao do tamanho dos pacotes:
ate 64 bytes : 14.857%
de 65 a 127 bytes : 25.251%
de 128 a 255 bytes : 43.230%
de 256 a 511 bytes : 1.644%
de 512 a 1023 bytes : 3.596%
de 1024 a 1518 bytes: 11.422%
Eficiencia da linha: 65%
Colisoes: 0.003136
venus{analucia}52: !!
BAIXO : 143616.000000 bits/s 98166.130000
Throughput Baixo
Status da linha: NORMAL 14.361600%
Distribuicao do tamanho dos pacotes:
ate 64 bytes : 14.857%
de 65 a 127 bytes : 25.251%
de 128 a 255 bytes : 43.227%
de 256 a 511 bytes : 1.644%
de 512 a 1023 bytes : 3.596%
de 1024 a 1518 bytes: 11.423%
Eficiencia da linha: 65%
Colisoes: 0.003135
venus{analucia}53:


0
venus.inf.ufsc.br etherStatsPkts
venus.inf.ufsc.br etherStatsPkts
venus.inf.ufsc.br etherStatsPkts
venus.inf.ufsc.br etherStatsPkts
Feb 29 1996
Feb 29 1996


60
50
40
30
20
10
0
venus.inf.ufsc.br etherHistory
Feb 29 1996
Feb 29 1996


4000000
3500000
3000000
2500000
2000000
1500000
1000000
500000
0
venus.inf.ufsc.br etherStatsPkts
Feb 29 1996
Feb 29 1996


venus.inf.ufsc.br etherHistoryCollisions
7
6
5
4
3
2
1
0
venus.inf.ufsc.br etherHistoryCollisions
Feb 29 1996
Feb 29 1996